École doctorale IAEM Lorraine

Surface Realisation from Knowledge Bases

THÈSE

présentée et soutenue publiquement le 20 Janvier 2016

pour l'obtention du

Doctorat de l'Université de Lorraine

(Mention Informatique)

par

Bikash Gyawali

Composition du jury

Directrice :	Claire Gardent	Directrice de recherche, CNRS, LORIA, France
Rapporteurs :	Karin Harbush	Professor, Universität Koblenz-Landau, Germany
	Albert Gatt	Senior Lecturer, University of Malta, Malta
Examinateurs :	Christophe Cerisara	Chargé de recherche, CNRS, LORIA, France
	Guy Perrier	Professor Emerite, Université de Lorraine, France
	Patrick Saint-Dizier	Directeur de recherche, CNRS, IRIT, France (Président)

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Acknowledgements

I take this opportunity to express my sincere thanks to everyone who directly or indirectly contributed to the success of my PhD studies. First and foremost thanks to my supervisor, Claire. You guided me with enthusiasm and supported me in exploring new research directions. Under your supervision, I felt free to try and test my ideas (which were, at times, very random and naive) but had the privilege of relying on your very strong expertise for judgements. Your gentle but firm reminders helped me stay on track, produce good results and successfully complete my PhD studies. I enjoyed a fruitful, healthy and rich research experience with you and had ample opportunities to learn; thank you very much for all these.

My dear wife, Rupa, you deserve a very special credit for supporting me in times of good and bad, happy or sad. Living in a foreign country is already quite difficult; you have endured me throughout these years and supported me in every possible ways for completing my PhD. Thank you very much indeed. Special thanks to my parents and family members who have helped me to become what I am today.

Thank you jury members, Albert Gatt, Christophe Cerisara, Guy Perrier, Karin Harbush and Patrick Saint-Dizier for reviewing my research and providing suggestions for improving the manuscript.

My colleagues in the SyNaLP team – Ali, Alejandra, Alexandre, Celine, Christophe, Ingrid, Laura, Lina, Shashi – thank you everyone for forming a very cooperative working environment and giving me numerous feedbacks in my work. The team meetings, presentations and discussions we had from time to time helped me to improve upon my work.

Many thanks to Imen, Laura, Mariem, Nabil and Nicolas for helping me write the French summary section of this manuscript. Thank you Guillaume for suggesting presentation skills for the defense session.

The *petit* Nepalese community in Nancy consisting of hardly 3-4 persons at any time – Anju, Binod, Jugdish, Manish, Nirmal, Santosh, Sharib – thank you all for the good times spent together.

Un grand merci to Université de Lorraine and the LORIA research laboratory for providing me a sound research platform and helping me realise the PhD dream.

Résumé

Bases de Connaissances et Réalisation de Surface
Bikash Gyawali

La Génération Automatique de Langue Naturelle (GLN) vise à produire des textes ou de la parole dans une langue humaine à partir d'un ensemble de données non-linguistiques. A partir d'un ensemble de données et d'un but communicatif, la génération automatique de textes exécutera trois sous-tâches principales: (i) sélection et organisation d'un sous-ensemble de données d'entrée, lesquelles répondent au but communicatif; (ii) détermination des mots à utiliser pour verbaliser les données d'entrée; et (iii) regroupement de ces mots en un texte en langue naturelle verbalisant les données sélectionnées. La dernière sous-tâche est connue comme la tâche de Réalisation de Surface (RS). Dans ce travail de thèse, nous étudions la tâche de réalisation de surface quand les données d'entrée sont extraites de Bases de Connaissances (BC). Ce qui motive la verbalisation de bases de connaissances est le besoin, d'une part, de faciliter l'accès au contenu de celles-ci (motif pratique) et, d'autre part, de développer des méthodes générales et fondées sur des principes linguistiques (motif théorique).

Nous proposons deux nouvelles approches pour la réalisation de surface à partir de bases de connaissances: une approche supervisée et une approche faiblement supervisée.

Approche supervisée: La Tâche Commune de Réalisation de Surface KBGen (KBGen challenge) a été conçue avec le but de comparer et d'évaluer les systèmes de réalisation de surface prenant en entrée des bases de connaissances. A partir d'un sous-ensemble de données cohérent extrait de la base de connaissances, l'objectif de la tâche de réalisation de surface est de produire des phrases complexes en anglais qui sont à la fois grammaticales et naturelles. Dans cette tâche commune, le challenge met à disposition des participants un petit (208 exemples) corpus parallèle de paires phrase / sous-ensemble de données de la base de connaissances ainsi que des lexiques qui associent les symboles de la base de connaissances à des mots et à des phrases. Dans la première partie de cette thèse, nous présentons une méthode pour extraire une Grammaire d'Arbres Adjoints basée sur les traits (Feature Based Lexicalized Tree Adjoining Grammar (FB-LTAG)) à partir d'un corpus parallèle de textes et de données. La grammaire FB-LTAG résultante inclut une sémantique compositionnelle basée sur l'unification et peut être utilisée par un réalisateur de surface existant pour produire des phrases à partir de bases de connaissances. Nous appliquons la méthode sur les données de KBGen, nous étendons le réalisateur de surface existant en ajoutant un mécanisme de classement ainsi que de recherche en faisceau, et nous testons la grammaire obtenue sur les données KBGen. Les évaluations expérimentales montrent que notre approche est

plus performante qu'une approche guidée par les données qui s'appuient sur une grammaire probabiliste extraite automatiquement et que les phrases produites s'approchent de celles produites avec une grammaire symbolique développée manuellement. En outre, une caractéristique de notre approche est qu'elle s'appuie sur une grammaire compacte (quelques centaines d'arbres) et fondée sur des principes linguistiques (elle suit les principes sémantiques et de domaine de localité étendu dans les grammaires TAG). Nous montrons comment cette caractéristique donne lieu à une approche hybride où une grammaire extraite automatiquement peut être révisée manuellement et améliorer la couverture et la qualité des phrases produites.

Approche faiblement supervisée: Une limitation importante de l'approche supervisée décrite précédemment est qu'elle requiert l'existence d'un corpus parallèle alignant un fragment de la base de connaissances avec une phrase verbalisant ce fragment. Dans la seconde partie de cette thèse, nous explorons par conséquent une approche pour la réalisation de surface à partir de données des base de connaisences qui utilise un lexique fourni mais ne requièrent pas ce type de corpus parallèle. A la place, nous construisons un corpus à partir de sources hétérogènes de textes liées au domaine des bases de connaissances pour lesquelles la réalisation de surface est développée (dans ce cas, biologie) et nous utilisons ce corpus pour identifier les lexicalisations possibles des symboles de la BC (classes et relations). Nous utilisons ensuite ce corpus pour estimer les probabilités des lexicalisations des symboles de la BC, des cadres de sous-catégorisation et des liens entre les différents arguments syntaxiques et sémantiques d'un évènement donné. Nous proposons des modèles probabilistes pour la sélection de cadres de sous-catégorisation et associations syntaxiques/sémantiques appropriés et nous utilisons une fonction attribuant un score qui utilise les probabilités pour verbaliser une entrée donnée. Nous présentons des évaluations automatiques et des évaluations réalisées par les humains des phrases générées et nous analysons les problèmes lié à l'apprentissage automatique à partir d'un corpus non-aligné.

Dans chacune de ces approches, nous utilisons des données dérivées d'une ontologie biomédicale existante comme référence d'entrée (à savoir la base de connaissances AURA [Chaudhri *et al.*, 2013]). Cependant, nos méthodes sont génériques et peuvent être facilement adaptées pour une entrée à partir d'autres ontologies pour lesquels un corpus parallèle/non-parallèle existe.

Abstract

Surface Realisation from Knowledge Bases
Bikash Gyawali

Natural Language Generation (NLG) is the task of automatically producing natural language text to describe information present in non-linguistic data. Given some non-linguistic data as input and a defined communicative goal, NLG involves three main tasks: (i) selecting and structuring the relevant portion of input data which addresses the specified communicative goal; (ii) determining the words that will be used to verbalise the selected data; and (iii) mapping these words into a natural language text verbalising the information contained in the selected data. The latter task is known as Surface Realisation (SR) and in this thesis, we study the SR task in the context of input data coming from Knowledge Bases (KB). The motivation for verbalising KB data comes from the need of having human friendly access to such data (practical motive) and of developing generic and linguistically principled approaches for doing so (theoretical motive).

We present two novel approaches to surface realisation from knowledge base data: a supervised and a weakly supervised approach.

Supervised Approach: The KBGen challenge [Banik *et al.*, 2012, Banik *et al.*, 2013] was designed to compare and evaluate surface realisation systems taking as input knowledge base data. Given a knowledge base fragment which forms a coherent unit, the task was to generate complex sentences which are both grammatical and fluent in English. The challenge made available to the participants a small (207 training examples) parallel corpus of text and KB fragment pairs as well as lexicons mapping KB symbols to words and phrases. In the first part of this thesis, we present a corpus-based method for inducing a Feature Based Lexicalized Tree Adjoining Grammar (FB-LTAG) from a parallel corpus of text and data. The resulting extracted TAG includes a unification based semantics and can be used by an existing surface realiser to generate sentences from KB data. We apply our induction method to the KBGen data, use an existing surface realiser and implement a ranking module to test the resulting grammar on KBGen test data. Experimental evaluation shows that our approach outperforms a data-driven generate-and-rank approach based on an automatically induced probabilistic grammar; and yields results that are close to those produced by a handcrafted symbolic approach. Moreover, a distinguishing feature of our approach is that it relies on an automatically extracted grammar that is compact (a few hundred trees) and linguistically principled (it follows the semantic and extended domain of locality principles of Tree Adjoining Grammar). We show that this feature allows for a hybrid approach where an automatically extracted grammar can be manually revised to improve both coverage and output quality.

Weakly Supervised Approach. A strong limitation of the supervised approach just described is that it requires the existence of a parallel corpus aligning a KB fragment with a sentence verbalising that fragment. In the second part of this thesis, we therefore explore an approach for surface realisation from KB data that uses a supplied lexicon but does not require a parallel corpus. Instead, we build a corpus from heterogeneous sources of text related to the domain of the knowledge base for which surface realisation is being developed (in this case, biology) and we use this corpus to identify possible lexicalisations of the KB symbols (classes and relations). We then use this corpus to estimate the probabilities of KB symbol lexicalisations, of subcategorisation frames and of the linking between the various syntactic and semantic arguments of a given event. We propose probabilistic models for the selection of appropriate frames and syntax/semantics mapping and use a scoring function that utilises the learnt probabilities to verbalise the given input. We present automatic and human based evaluations of the output sentences and analyze issues relevant to learning from non-parallel corpora.

In both these approaches, we use the KBGen data as a reference input. The KBGen data is itself derived from an existing biomedical ontology (namely, the AURA Knowledge base, [Chaudhri *et al.*, 2013]). Our methods are generic and can be easily adapted for input from other ontologies for which a parallel/non-parallel corpora exists.

Je dédie cette thèse à mon temps passé à Nancy.

Contents

Bases de Connaissances et Réalisation de Surface

Sommaire

1 Introduction

1.1 Génération automatique de langue naturelle et réalisation de surface(RS)

La Génération Automatique de Langue Naturelle (GLN) peut être définie comme la tâche qui consiste) produire un texte en langue naturelle à partir d'informations codées dans un système de représentation machine (par exemple: les bases de données, les bases de connaissances, les formules logiques, etc.). La représentation et le stockage des informations dans de tels systèmes sont souvent régis par des contraintes formelles, ce qui les rend difficiles à utiliser pour les humains. La GLN se charge de générer des descriptions textuelles de ces informations en langue naturelle (telles que le français, népalais, etc.) et offre ainsi un moyen naturel et fluide de communication pour les utilisateurs humains.

[Reiter and Dale, 2000] présente une architecture de référence pour les systèmes GLN. Il s'agit d'une architecture en pipeline qui modélise la tâche de génération en trois tâches séquentielles: Planification de Document, Planification de Surface et Réalisation de Surface. La planification de document est la tâche d'identification et de structuration des unités pertinentes à partir de l'entrée. La planification de surface (appelé aussi micro-planification) se base sur cette tâche et explore ainsi l'aspect linguistique des unités sélectionnées, comme par exemple, en identifiant les formes de mots corrects pour décrire une entité dans l'entrée ou la détermination des unités de contenu qui se réfèrent à l'autre. La réalisation de surface (RS) se charge de produire des expressions de surface réelles qui verbalisent ces unités de contenu dans des structures syntaxiquement correctes et sémantiquement cohérentes.

1.2 Entrées de RS

Selon l'application traitée, la nature des entrées de la réalisation de surface (RS) diffère. Trois grands types d'entrées sont généralement distingués. Le premier type comprend une collection de données, soit des données brutes non structurées (connues aussi sous le nom de "données plates), soit des données organisées par leurs interrelations dans un ensemble d'enregistrements comme dans une base de données. Des exemples de tels entrées RS ont été présentés dans [Reiter *et al.*, 2005], [Belz, 2007], [Ratnaparkhi, 2000], [Angeli *et al.*, 2010], [Konstas and Lapata, 2012b], etc. Le deuxième type d'entrées comprend les représentations linguistiques (syntaxiques ou bien sémantiques), comme par exemple, les arbres de dépendance et les représentations de discours [Bohnet *et al.*, 2010], [Wang and Zhang, 2012], [Kondadadi *et al.*, 2013], [Dethlefs and Cuayáhuitl, 2012a] etc. Enfin, le troisième type est constitué des structures provenant de divers formalismes logiques, comme la logique de premier ordre [Gerdemann and Hinrichs, 1990], les termes lambda [Lu and Ng, 2011] et les bases de connaissances [Stevens *et al.*, 2011], [Cimiano *et al.*, 2013], [Ell and Harth, 2014], [Duma and Klein, 2013], etc.

1.3 Approches de RS

Dans la littérature, trois types d'approches ont été distinguées pour la réalisation de surface : i) approches basée sur les chablons; ii) approches basées sur les grammaires et iii) approches d'associations directes. Dans les approches basées sur les chablons, les structures linguistiques incomplètes contenant des trous sont utilisées. Pour une réalisation de surface réussite, tous les trous dans les chablons doivent êtres remplis à partir des données d'entrée, et ainsi générer une phrase. Plusieurs approches de réalisation de surface basées sur les chablons ont été proposées : à l'aide des chablons définies à la main [Van Deemter and Odijk, 1997], [McRoy *et al.*, 2003], [Androutsopoulos *et al.*, 2013], etc., ou des chablons extraits automatiquement à partir d'un corpus d'un domaine particulier [Duma and Klein, 2013], [Ell and Harth, 2014], [Cimiano *et al.*, 2013], etc.. Dans les approches basées sur les grammaires, une grammaire est utilisée pour décrire l'association entre les données d'entrée et l'expression de surface. Une grammaire est un ensemble de règles spécifiant la relation entre les fragments de l'entrée, les constituants syntaxiques et les expressions de la langue naturelle. Pour la génération, les règles sont combinées selon les contraintes imposées par la grammaire afin d'obtenir un texte en sortie. Plusieurs travaux suivent cet axe en utilisant : i) une grammaire spécifique à une tâche [Elhadad, 1993], ii) une grammaire existante à large couverture [Carroll and Oepen, 2005], [Rajkumar *et al.*, 2011], [Cahill and Van Genabith, 2006], [Narayan and Gardent, 2012b], et iii) une grammaire créée automatiquement par des méthodes d'apprentissage automatique [Lu and Ng, 2011], [Belz, 2007], [DeVault *et al.*, 2008b]. Enfin, les approches d'association directe transforment directement les variables d'entrées en des expressions de surface à partir d'un corpus parallèle sans considérer les relations syntaxiques et sémantiques [Bohnet *et al.*, 2010], [Wang and Zhang, 2012], [Ballesteros *et al.*, 2015], [Guo *et al.*, 2008], [Filippova and Strube, 2007], [Ringger *et al.*, 2004], [Zhong and Stent, 2009], [Konstas and Lapata, 2012b], [Wong and Mooney, 2007] etc.

1.4 RS à partir des bases de connaissances

Dans cette thèse, nous présentons notre travail sur la RS à partir d'une base de connaissances (ontologie).

Une ontologie est une conceptualisation d'un domaine à travers des entités ontologiques, à savoir les concepts, les individus, les relations et les axiomes. Notre choix d'ontologies pour la RS est motivé par des raisons théoriques et pratiques. D'un point de vue théorique, ça nous aide à étudier les défis de la verbalisation posés par les expressions logiques des ontologies et à explorer les méthodes qui les résolvent. D'un point de vue pratique, utiliser les ontologies permet de profiter de toutes les technologies, les corpus, les ressources, et les outils proposés par la communauté de web sémantique.

Notre choix d'adopter une approche basée sur une grammaire pour la réalisation de surface est inspiré par des motivations linguistiques. Les deux autres types d'approches (à base de chablons ou bien d'associations directes) discutées précédemment ne peuvent pas modéliser les contraintes linguistiques (syntaxiques et sémantiques) qui régissent la bonne forme des phrases dans une langue quelconque. Ainsi, dans ces approches, les contraintes linguistiques simples (e.g. l'accord sujet-verbe) et les contraintes les plus complexes (e.g. la liaison entre les arguments syntaxiques et sémantiques) sont tout simplement ignorées. Cependant, dans les approches basées sur la grammaire, les données d'entrée sont associées à des constructions linguistiques précisant leurs rôles syntaxiques, tels que les informations des cadres de sous-catégorisations et catégories lexicales. La grammaire décrit un ensemble de règles syntaxiques qui stipulent la combinaison des constituants syntaxiques (sélectionnés par les données d'entrée) pour la production de la verbalisation correcte de la totalité de l'entrée. De cette façon, une approche basée sur la grammaire modélise la relation syntaxique parmi les données d'entrée et fournit un chablon linguistique de la réalisation de surface.

En outre, pour éviter les problèmes associés aux grammaires construites manuellement (perte de temps et intervention massive d'utilisateurs) ou à la réutilisation des grammaires à large couverture (difficultés de conversion de format [Callaway, 2003] et [Busemann, 1996]), nous proposons un apprentissage automatique de la grammaire à partir d'un corpus d'un domaine donné. Comme le résument les Sections 2 et 3 et le détaillent les Chapitres 3 et 4, nous développons une approche pour apprendre une grammaire à partir d'un corpus parallèle et non parallèle. Le chapitre 3 présente notre approche supervisée pour apprendre une grammaire associant entrée et réalisation de surface à partir d'un corpus parallèle à l'aide d'un lexique fourni a priori. Le chapitre 4 présente une autre approche pour la réalisation de surface basée sur une grammaire qui utilise un lexique fourni mais qui ne nécessite pas un corpus parallèle.

2 Verbalisation de triples – Une approche supervisée

Dans le contexte de la réalisation surface à partir de bases de connaissances, les ressources créées manuellement (chablons ou grammaires) ont été largement utilisées. Des travaux précédents, par exemple [Carenini *et al.*, 1994], [Paris, 1988], [Aguado *et al.*, 1998], [Galanis *et al.*, 2009] utilisent des chablons écrits manuellement pour établir la correspondance entre le texte et l'information sémantique dans les bases de connaissances. D'autres travaux tels que [Bontcheva and Wilks., 2004], [Williams and Power, 2010] et [Cimiano *et al.*, 2013] utilisent des règles spécifiées manuellement.

Ici nous explorons un approche alternative dans laquelle nous induisons, à partir d'un corpus parallèle alignant texte et données, une grammaire qui sera utilisé par un réalisateur de surface. Étant donné un ensemble d'apprentissage constitué de paires $(\{t_1, \ldots, t_n\},S)$ où $\{t_1, \ldots, t_n\}$ est un ensemble de triplets issus d'ontologies, et S est une phrase verbalisant cet ensemble de triplets, nous développons une méthodologie pour l'apprentissage de grammaires TAG, qui capture la correspondance entre les triplets des bases de connaissance et le texte. De plus, nous connectons automatiquement les unités sémantiques de l'entrée aux constructions syntaxiques de la grammaire extraite et imposons ainsi une intégration synaxique et sémantique plus forte en utilisation une sémantique basée unification ([Gardent and Kallmeyer, 2003]). La méthode d'induction suit des principes linguistiques et permet d'obtenir une grammaire compacte et facilement généralisable qui permet de traiter d'entrée non vues dans le corpus d'apprentissage.

```
:TRIPLES (
   (|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
   (|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
   (|Gated-Channel64605|  |has-function| |Release-Of-Calcium646|)
   (|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
:INSTANCE-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Release-Of-Calcium|)
   (|Particle-In-Motion64582| |instance-of| |Particle-In-Motion|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Endoplasmic-Reticulum|)
   (|Gated-Channel64605| |instance-of| |Gated-Channel|))
:ROOT-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Event|)
   (|Particle-In-Motion64582| |instance-of| |Entity|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Entity|)
   (|Gated-Channel64605| |instance-of| |Entity|))
```

Sentence :
The function of a gated channel is to release particles from the endoplasmic reticulum.

Figure 1: Un exemple d'apprentissage avec le dataset KBGen

Notre entrée (le dataset KBGen) est constitué de données issues d'une base de données connaissances biologique existante (la KB Bio 101 [Chaudhri *et al.*, 2013]) fournie par le défi KBGen [Banik *et al.*, Banik *et al.*, 2012, 2013]. L'objectif de ce défi est d'évaluer la génération de grammaires à partir de bases de connaissances. Comme [Angeli *et al.*, 2010], nous utilisons le terme "scénario" pour décrire le contenu d'une base de connaissance associé avec la phrase qui lui correspond. Le dataset KBGen est constitué de 207 scénari d'apprentissages (un exemple est décrit dans la Figure1), et de 72 scenari de test. Nous apprenons une grammaire qui connecte le contenu de la base de connaissance aux chaines de caractères des scenari d'apprentissage, et utilisons les phrases dans les scenari de test comme phrase de référence pour l'évaluation des phrases générées par notre approche.

Le dataset KBGen fournit également un lexique qui liste les mots et les phrases qui peuvent être utilisés pour verbaliser les variables (entités et évènements) apparaissant dans les ensembles d'apprentissage et de test. Le lexique définit un mapping entre types d'évènements, verbes, leur formes fléchies et leur nominalisation ainsi qu'entre entité, noms et forme plurielle. Par exemple, les entrées du lexiques pour les entités et évènements présentés Figure 1 sont décrites dans la Figure 2 ci-dessous.

`Release-Of-Calcium`	releases, release, released, release
`Particle-In-Motion`	molecule in motion, molecules in motion
`Endoplasmic-Reticulum`	endoplasmic reticulum, endoplasmic reticulum
`Gated-Channel`	gated channel, gated channels

Figure 2: Example Entries from the KBGen Lexicon

Nous avons pour but d'apprendre les arbres dune grammaire d'arbres adjoints lexicalisée à traits (Feature-Based Lexicalised Tree Adjoining Grammar, FB-LTAG) chaque exemple de l'ensemble d'apprentissage, afin de construire une grammaire qui pourra ensuite être utilisée pour la génération à partir des données de test. À cette fin, nous traitons chaque scenario d'apprentissage séparément, la tâche pouvant être résumée comme suit. Premièrement, pour chaque scenario d'apprentissage, nous alignons les variables de la base de connaissances aux chaines de caractères de la phrase correspondante, en utilisant la correspondancde exacte ou presque exacte d'une ou plusieurs entrées lexicales. Ensuite, nous définissons un ensemble d'heuristiques et les utilisons pour projeter les variables de la base de connaissance sur les noeuds syntaxiques de l'arbre syntaxique de la phrase. Une fois toutes les projections effectuées, nous obtenons les sous-arbres ayant pour racines les noeuds de projection des variables et nous les associons aux variables qui correspondent sémantiquement. Les motivations et les étapes pour extraire de tels arbres sont expliqués en détails dans la Section 3.5.2.

Deuxièmement, nous découpons les arbres obtenus pour les variables évènementielles en des arbres plus petits, chacun représentant des arguments syntaxiques ou sémantiques à un moment donné. Comme détaillé dans la Section 3.5.4, ceci permet la création de nouveaux arbres TAG auxiliaires qui représentent des verbalisations de relations sémantiques indépendantes (servant couramment de modifieurs optionnels dans les phrases). En outre, cette stratégie permet de limiter un overfit de la grammaire et de traiter de cas d'entrée où la combinaison des relations donnée n'a pas été vue dans les données de test.

Enfin, pour traiter les cas de test avec des variables d'évènement et d'entités non detectées, nous définissons une procédure d'adaptation de la grammaire automatique qui connecte un arbre existant dans la grammaire à l'entrée de test donné, en se basant sur leur similarité sémantique. Dans la Section 3.5.5, nous montrons que cette procédure fournit une couverture complète pour les 72 scenari de test, et que nous pouvons adapter les arbres issus de différents ensembles d'apprentissage pour traiter la même entrée de test.

Pour faire la génération avec la grammaire extraite, nous utilisons un réalisateur de surface existant, GenI [Gardent *et al.*, 2007]. Nous évaluons les phrases générées en les comparant avec deux systèmes différents qui génèrent à partir du même dataset KBGen – le système UDEL [Butler *et al.*, 2013] (un système construit manuellement avec un système de règles) et le système IMS [Zarrieβ and Richardson, 2013] (un système statistique utilisant une grammaire probabiliste). Les phrases écrites par l'humain pour chaque scenario de test fournis par le dataset KBGen servent de références pour comparer les décisions de chaque système. Une évaluation automatique est réalisé en terme de score BLEU [Papineni *et al.*, 2002] et une évaluation par des humains a été menée sous la forme d'une enquête demandant à des utilisateurs humains de noter les phrases générées par chaque système pour leur facilité de compréhension (le text est il facile à lire?), leur qualité grammaticale (la phrase est elle naturelle et bien formée?) et leur adéquation sémantique avec les phrases de référence (Le sens de la phrase générée est il le meme que celui de la phrase de référence?). L'analyse de ces deux évaluations montrent que notre système offre de moins bonnes performances que le système UDEL construit manuellement, mais de meilleures performances que le système statistique IMS.

Ainsi, en utilisant le dataset KBGen nous proposons une nouvelle méthode pour extraction automatique de grammaires qui peuvent permettre de connecter sémantiquement et syntactiquement les triplets des bases de connaissances avec les chaînes de caractères d'un texte. La grammaire résultante est constituée via des principes linguistiques et possède des performances statisfaisantes au regard des autres approches symboliques et statistiques. Par ailleurs, notre méthode est générique et peut être adaptée à toute base de connaissance.

3 Verbalisation des événements n-aire dans les Ontologies – Une approche faiblement supervisée

Une forte limitation de l'approche supervisée qu'on vient de décrire est qu'elle nécessite l'existence d'un corpus parallèle alignant un fragment KB avec une phrase qui verbalise ce fragment. Dans la deuxième partie de cette thèse, nous explorons donc une approche pour la réalisation de surface à partir des données KB qui utilise un lexique fourni mais ne nécessite pas un corpus parallèle.

Notre entrée pour cette expérience, le KBGen$^+$ dataset, est dérivée du KBGen dataset discuté plus tôt. Dans le KBGen dataset, l'entrée est composée des unités de contenu, dont chacune exprime un ensemble de relations entre les types de concepts différents, à savoir événement-à-entité, événement-à-événement, entité-à-événement, entité-à-entité et les relations propriétés-valeurs. Cependant, dans ce travail, nous nous intéressons à décrire les événements en lien avec leurs arguments de type entité seulement et, par conséquent, nous traitons le KBGen dataset pour produire tous les fragments KB qui représentent un événement unique avec des rôles à des entités seulement. Le KBGen$^+$ dataset est donc une collection de descriptions d'événements biologiques par lesquels une *description d'événement* est constituée d'un événement, ses arguments et les n-plusieurs rôles reliant chaque argument à l'événement. Au total, nous obtenons 336 descriptions d'événements pour notre KBGen$^+$ dataset. Un exemple KBGen$^+$ entrée (seulement le *:TRIPLES* section) dérivée de l'entrée KBGen de la Figure 1 est montrée dans la Figure 3 ci-dessous. Notez que nous créons les phrases de référence pour notre KBGen$^+$ dataset en ne retenant que les structures de description de l'événement dans les phrases correspondantes du KBGen avec des modifications minimales et analyse manuelle minutieuse.

```
:TRIPLES (
   (|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
   (|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
   (|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
```

Sentence :
A gated channel release particles from the endoplasmic reticulum.

Figure 3: Un Exemple de Scénario d'Apprentissage à partir du KBGen$^+$ Dataset

Pour générer automatiquement des verbalisations en langage naturel des descriptions d'événements dans le répertoire KBGen$^+$ dataset, nous proposons une méthode probabiliste qui extrait les possibles cadres de verbalisation à partir d'un large corpus d'un domaine spécifique en biologie et qui utilise les probabilités à la fois pour sélectionner un cadre approprié étant donnée une description de l'événement et pour déterminer la mappage entre les arguments syntaxiques et sémantiques. À cette fin, nous commençons par la collecte des phrases provenant de plusieurs corpus du domaine biomédical publiquement disponibles. Ceci inclu les corpus BioCause [Mihăilă *et al.*, 2013], BioDef[1], BioInfer [Pyysalo *et al.*, 2007], Grec [Thompson *et al.*, 2009], Genia [Kim *et al.*, 2003a] and PubMedCentral (PMC)[2]. Nous incluons aussi les phrases disponibles dans les des concepts nommés dans l'ontologie KB Bio 101 . Cette collection personnalisée de phrases sera le corpus sur lequel notre approche d'apprentissage va se construire.

Pour identifier les phrases du corpus qui pourraient contenir des verbalisations des événements et entités du KBGen$^+$, nous avons également besoin d'un lexique mappant les variables événement et entité contenues dans KBGen$^+$ à des mots ou des phrases langage naturel. Pour cela, nous prenons le lexique fourni par le défi KBGen et nous l'augmentons avec les entrées synonymes pour les événements et entités du KBGen$^+$ trouvés dans Mesh[3] dans le vocabulaire BioDef. Mesh est un dictionnaire existant à large couverture des termes dans les sciences de la vie et fournit la synonymie des termes. BioDef est notre nom personnalisé pour vocabulaire de synonymes que nous construisons automatiquement en analysant les entrées dans la section ⟨Synonyms⟩ des pages html rampés d'un dictionnaire de biologie ouvert à http://www.biology-online.org/dictionary/. The lexique résultant est donc une fusion de toutes les entrées extraites de toutes les sources mentionnées ci-dessus pour tous les événements et entités du KBGen$^+$.

Equipés avec les phrases et le lexique, nous procédons à extraire les cadres syntaxiques pour les événements survenus dans le dataset KBGen$^+$ tout en traitant chaque événement à son tour. Pour chaque événement e dans le dataset KBGen$^+$ nous recherchons toutes les phrases S dans le corpus qui mentionnent une ou plusieurs des formes de mot disponibles pour cet événement dans le lexique fusionné. Chacune de ces phrases $s \in S$ est ensuite analysée selon l'analyseur de dépendance Stanford [4] pour la structure de dépendance effondrée. Depuis l'arbre d'analyse de dépendance résultant, nous extrayons la sous-arborescence t enracinée au nœud étiqueté avec la forme de mot pour la variable d'événement et couvrant seulement ses dépendances immédiats en charge (ie les nœuds enfants directs). Le cadre obtenu pour l'événement e depuis cette phrase s est alors une chaîne composée de séquence or-

[1] Obtenu par analyse de la section ⟨Supplement⟩ des pages html rampé à partir de http://www.biology-online.org/dictionary/

[2] ftp://ftp.ncbi.nlm.nih.gov/pub/pmc

[3] http://www.nlm.nih.gov/mesh/filelist.html

[4] http://nlp.stanford.edu/software/lex-parser.shtml

donnée de relations de dépendance se produisant dans t ainsi que le tag partie du discours (pos) du nœud racine. Dans le cadre, nous généralisons les tags NN, NNS, NNP et NNPS comme NP; les pos tags VBD, VBG, VBN, VBP et VBZ comme VB et nous gardons le reste tel qu'il est. Par exemple, étant donné la phrase et son arbre d'analyse de dépendance correspondant comme indiqué dans 4, un cadre VB enraciné *nsubj, VB, dobj* est obtenu pour l'événement Block indiquant que la forme du verbe *block* exige un sujet et un objet.

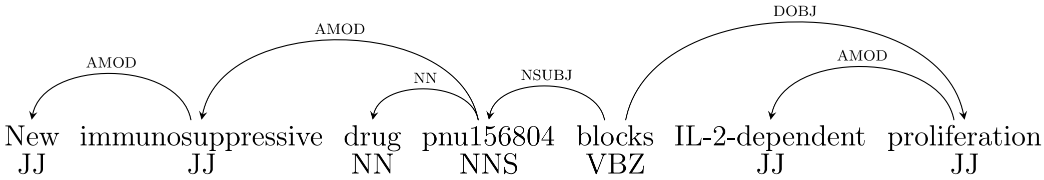


Figure 4: Exemple d'arbre de dépendances

La procédure d'extraction de cadre peut nous fournir une grande variété de modèles de verbalisation syntaxiques pour chaque événement dans la dataset KBGen$^+$. De plus, nous devons établir la correspondance syntaxique/sémantique entre la structure syntaxique dans des cadres et les rôles sémantiques dans la dataset KBGen$^+$ pour un système réussi. Pour résoudre ces problèmes, nous proposons trois différents modèles probabilistes qui sont entrainés sur les cadres extraites et seront utilisés pour générer les descriptions des événements de KBGen$^+$ pendant la phase de test. Étant donné F un ensemble de structures syntaxiques, E un ensemble de événements de KBGen$^+$ D un ensemble de noms de dépendance syntaxiques et R, un ensemble de KB rôles, nous construisons trois modèles probabilistes génératifs, à savoir la $P(f|e)$, $P(f|r)$ et $P(d|r)$.

Le modèle $P(f|e)$ avec $f \in F$ et $e \in E$ dénote la probabilité d'un cadre sachant un événement. Elle est calculé comme suit:

$$P(f|e) = \frac{counts\left((f,e) \in \mathcal{C}_e\right) + 0.1}{\sum_{f'}\left(counts\left((f',e) \in \mathcal{C}_e\right) + 0.1\right)} \tag{1}$$

où $\mathcal{C}_e$ représente la collection de toutes les cadres extraites de l'événement e à partir du corpus de phrases; $counts(f,e)$ est le nombre de fois que le cadre f est observée pour l'événement e dans $\mathcal{C}_e$ et $counts(f',e)$ est la fréquence de tout cadre f' observé pour l'événement e dans $\mathcal{C}_e$.

Le modèle $P(f|r)$ avec $f \in F$ et $r \in R$ dénote la probabilité d'un cadre sachant un rôle. Elle est calculée comme suit:

$$P(f|r) = \frac{counts\left((f, r) \in \mathcal{C}_r\right) + 0.1}{\sum_{f'} \left(counts\left((f', r) \in \mathcal{C}_r\right) + 0.1\right)} \tag{2}$$

où $\mathcal{C}_r$ représente la collection de tous les cadres alignés sur le rôle r; $counts(f, r)$ est le nombre de fois que le cadre f est observé pour le rôle e dans $\mathcal{C}_r$ et $counts(f', r)$ est la fréquence de tout cadre f' observé pour le rôle r dans $\mathcal{C}_r$. Pour ce modèle, nous supposons qu'un cadre de l'événement f extrait de certaines sous-arborescence de dépendances t est aligné à un rôle r dans le total des descriptions d'événements KBGEN$^+$ chaque fois que t a une entité e comme dépendante et l'entité e est lié via le rôle r dans l'une des descriptions d'événements dans la dataset KBGEN$^+$.

Le modèle $P(d|r)$ vise à apprendre la relation syntaxe/sémantique pour verbaliser le cadre sélectionnée. Il est calculé comme suit:

$$P(d|r) = \frac{counts\left((d, r) \in \mathcal{C}_d\right) + 0.1}{\sum_{d'} \left(counts\left((d', r) \in \mathcal{C}_d\right) + 0.1\right)} \tag{3}$$

où $\mathcal{C}_d$ représente la collection de toutes les dépendances alignées sur le rôle r; $counts(d, r)$ est le nombre de fois la dépendance d est observée pour le rôle r dans $\mathcal{C}_d$ et $counts(d', r)$ est la fréquence de toute dépendance d' observé pour le rôle r dans $\mathcal{C}_d$. Pour ce modèle, nous supposons qu'une relation de dépendance d présente dans une sous-arborescence de dépendances t peut être aligné à un rôle r dans le total des entrées à chaque fois que d lie une entité e en t et l'entité e est lié via le rôle r dans l'une des descriptions d'événements de l'entrée. A l'opposition du modèle '$P(f|r)$, ici, nous alignons les rôles de l'entrée à la relation de dépendance correspondante dans le cadre plutôt qu'au cadre lui-même.

La tâche de la réalisation de surface pour verbaliser les descriptions d'événements dans la base KBGEN$^+$ rend l'utilisation des modèles appris jusqu'ici. Etant donnée une description de l'événement, nous identifions d'abord l'événement e et l'ensemble des rôles $r_1 \ldots\ldots r_n$ qu'il contient. Nous définissons l'arité de l'événement e comme étant le nombre de types de rôles distincts présents dans la description de l'événement. Puis, à partir de tous les cadres présents dans $\mathcal{C}_e$ pour cet événement, nous ne sélectionnons que ceux qui ont la même arité (ceci est égal au nombre de dépendances syntaxiques dans le cas de cadres) que l'événement d'entrée. Tous ces cadres sont des cadres candidats pour notre tâche de réalisation de surface. Nous voulons identifier les 5 meilleurs cadres pour lesquels nous considérons deux fonctions de score alternatives (M1) et (M2).

$$P(f|e) \times \prod_{i=1}^{n} P(f|r_i) \tag{M1}$$

$$P(f|e) \times \prod_{i=1}^{n} P(f|r_i) \times \prod_{i=1}^{n} P(d_i|\hat{r}_i^f) \tag{M2}$$

où

$$(\hat{r}_1^f, \ldots, \hat{r}_n^f) = \underset{(s_1,\ldots,s_n)\in\mathcal{P}(\{r_1,\ldots,r_n\})}{\text{argmax}} \prod_{i=1}^{n} P(d_i|s_i)$$

et $\mathcal{P}(\{r_1, \ldots, r_n\})$ est l'ensemble des permutations de roles.

Nous sélectionnons les 5 cadres ayant les meilleurs scores (à partir des deux équations (M1) et (M2)) et de déterminer la correspondance entre dépendances syntaxiques que contient le cadre et les rôles sémantiques dans la description de l'événement d'entrée pour lequel ce cadre a été sélectionné en utilisant la fonction $(\hat{r}_1^f, \ldots, \hat{r}_n^f)$ défini ci-dessus. Une fois une telle cartographie est connue, la tâche de génération est réduite à remplir chaque fente de dépendance dans le cadre de l'entrée avec l'entité lexicale liée du rôle correspondant (l'entité argument) dans l'entrée et préserver le premier mot verbalisant l'évènement.

Nous évaluons les résultats obtenus à la fois quantitativement (évaluation Automatique) et qualitativement (évaluation Humaine) et nous analysons les problèmes relatifs à l'apprentissage à partir de corpus non-parallèle.

4 Conclusion

Au total, nous proposons deux nouvelles approches motivées linguistiquement pour la réalisation de surfaces à partir à partir de base de connaissances. Nous utilisons une entrée échantillon provenant d'une ontologie biomédicale existante, mais nos approches sont génériques et peuvent être facilement étendues à d'autres ontologies. Pour chacune de nos approche, nous présentons une description détaillée des procédures impliquées, nous montrons les résultats obtenus nous les analysons. Nous identifions les cas de problèmes éventuels, nous présentons les visions linguistiques dans les causes des échecs et nous fournissons des indications pour une les prochaines recherches.

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Thesis Scope

This thesis is about generating sentences from Knowledge Base data. Using an existing Knowledge Base (the KBBio101 ontology [Chaudhri *et al.*, 2013]) as reference, we present novel approaches to Surface Realisation from Knowledge Base (KB) data that are generic and independent of a particular KB domain.

Surface Realisation is the task of automatically producing surface text from some sentence size input. In a typical Natural Language Generation (NLG) system, it constitutes the final phase of the generation task. The reference architecture for NLG system proposed by [Reiter and Dale, 2000] presents a pipeline architecture for text generation which models the generation task as three main activities in sequence – Content Planning, Micro Planning and Surface Realisation. Content Planning is related to choosing and organizing content units from the input that are relevant to the communicative goal of the NLG system. Micro Planning builds upon this by exploring the linguistic aspects of the selected content units; for example, identifying word forms to describe an entity in the input or determining the content units that refer to each other. Finally, Surface Realisation is responsible for producing actual surface expressions that verbalise those content units in syntactically correct and semantically coherent structures when put in the context of natural language text.

Consider a toy input (Figure 1.1) derived from a sample database of student records to see how the different modules making up the NLG system come into play while generating a simple sentence.

Academic Year	Number of Students	Pass %	Attendance %
2009	200	52	44
2010	200	68	56
2011	200	80	32

Figure 1.1: A toy database containing student records

Assuming that we want to have a description of academic performance of students in different years, the Content Planning module would select the attributes "Attendance %", "Pass %" and "Academic Year" and organize them in that order while leaving out the "Number of Students" attribute since it contains redundant information (it is the same throughout all the "Academic Year" and the "Pass %" and "Attendance %" already abstract over the actual numbers). After the relevant contents are selected, the Content Planning module may also decide on the choice of discourse rhetorics to link them (e.g. using "Despite" to express a contrast in selected content). The Micro Planning phase might then determine the proper word forms to express the attributes and attribute values, for example, chosing the lexical form “success” to denote the “Pass %” attribute and using gradable adjectives like “low” or “large” to represent values below/above a certain threshold, etc. Finally, the Surface Realisation module might map the output of the micro planning step into a sentence such as (1) .

(1) *Despite low attendance, a large number of students succeeded in 2011.*

In this thesis, we focus our research objectives on the surface realisation task alone and for this, we assume that the Content Planning and Micro Planning tasks on the KB have been carried out beforehand. In practice, we accept a fragment of the KBio101 ontology extracted by the KBGen organisers for the KBGen challenge [Banik *et al.*, 2012, Banik *et al.*, 2013] as input to our surface realisation task. In this dataset provided by the KBGen challenge, each input describes a coherent unit of semantic content which can be verbalised by a single, possibly complex sentence that is grammatical and meaningful and the set of content units express as many different relations and concepts of different semantic types (events, entities, properties etc.) as possible.

Figure 1.2 shows a sample KBGen input. As can be seen, the `:TRIPLES` section specifies a single connected unit (possibly a graph) of binary relations between KB entities (events or individuals), the `:INSTANCE-TYPES` section provides information on the semantic types of the entities mentioned in the `:TRIPLES` section and the `:ROOT-TYPES` section defines the data types of these entities. Basically, the appropriate content units for the generation of a, possibly complex, sentence have already been determined and the lexicalisation information is provided. What remains is the development of methods for generating surface text expressions from these content units, such as the sentence shown in (2) for the input shown in Figure 1.2.

(2) *The function of a gated channel is to release particles from the endoplasmic reticulum.*

We experiment and evaluate a supervised and a weakly supervised approach for surface realisation from such inputs, discuss their feasibility to address different KB inputs and provide pointers to further research directions. Learning from parallel data (supervised learning) allows for immediate analysis of problems faced during verbalisation since the parallel text serves as a quick reference upon which the output from our system can be compared against. Also because the supervised approach learns from parallel text, it promises on reflecting the updates which may occur in the parallel text (in future) into the sentences generated by the system. Learning from

```
:TRIPLES (
   (|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
   (|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
   (|Gated-Channel64605|  |has-function| |Release-Of-Calcium646|)
   (|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
:INSTANCE-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Release-Of-Calcium|)
   (|Particle-In-Motion64582| |instance-of| |Particle-In-Motion|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Endoplasmic-Reticulum|)
   (|Gated-Channel64605| |instance-of| |Gated-Channel|))
:ROOT-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Event|)
   (|Particle-In-Motion64582| |instance-of| |Entity|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Entity|)
   (|Gated-Channel64605| |instance-of| |Entity|)))
```

Figure 1.2: Example input from KBGen

non-parallel texts (weakly supervised learning), however, allows for a more general setting for learning of generation resources whereby the need of having/authoring the parallel text is eliminated.

1.2 Motivations

There are both practical and theoretical grounds for exploring surface realisation from Knowledge Base data.

Knowledge Bases are software artifacts for storing, processing and inferring human knowledge in a computational framework. As such, Knowledge Bases encode real world knowledge in terms of concepts and relations represented via logical axioms. This makes them well suited for computational representation and reasoning but counter-intuitively less insightful for a human user to understand. Moreover, the expressive complexity of Knowledge Bases is ever increasing (i.e. more complex logical formalisms are emerging); the Knowledge Bases keep on dynamically evolving (so as to reflect newly acquired knowledge over time) and they often embody a large body of domain knowledge. In this context, a verbalisation system allows for automatic expression of Knowledge Base data in the most natural and comprehensible way to human, can scale up to the changing nature of Knowledge Base content and provides the information in a piecemeal fashion as relevant to the communicative goal set by the human user.

Thanks to the Semantic Web vision, Knowledge Bases (ontologies) have gained huge popularity as knowledge modeling tool across several domains and the number of ontologies on the web has exploded in recent years. Ranging from the environmental domain (e.g. the SWEET[5] ontology) to biological processes (e.g. the BioPAX[6] ontology), linguistic descriptions (e.g. the GOLD[7] ontology) and several others, ontologies provide ample opportunities for verbalising data from heterogeneous sources. Existing research on verbalisation from ontologies have mainly been motivated by the following three application scenarios :

- Description and Summary generation : Several works target the description of fragments of ontology data (in the form of logical axioms expressed in OWL or RDF standards) in response to different use-cases. [Androutsopoulos *et al.*, 2013], for example, generate multi-lingual (in English and Greek), multi-sentences text describing classes and individuals in ontologies; [Duma and Klein, 2013] present a natural language generation system producin short text describing factual, non-temporal information of entities in DBpedia[8] and [Mellish and Pan, 2006] additionally take into account the knowledge inferred from logical consequences of axioms in ontologies while verbalising them. Similarly, research in aggregating logical axioms in ontologies for obtaining textual summaries has been presented in [Bontcheva, 2005], [Williams and Power, 2010] etc. Overall, such systems aim to reduce the work of domain experts by avoiding the need to author resource definitions by hand and to provide an easy access to the information content to casual users.

- Intelligent Tutoring systems : Verbalising Knowledge Base data with the goal of tutoring human users in achieving some pedagogical goals have been reported in the field of Computer-Aided Language Learning (CALL). In [Amoia *et al.*, 2012], for example, the Knowledge Base contents relevant to a given pedagogical goal are selected and verbalised to make up sentences posing as exercise questions to human users. Along these lines is the task of question/answer generation from ontologies. [Papasalouros *et al.*, 2008] generate multiple-choice questions from ontology, [Gyawali, 2011] generate short answer texts to factoid questions posed upon ontologies and [Lopez *et al.*, 2007] present the AquaLog system which derives answers to user queries from multiple ontologies.

- Human friendly interface : Following the idea presented in [Tennant *et al.*, 1989], [Hallett *et al.*, 2007] proposed the “Conceptual Authoring” model which

[5] `http://sweet.jpl.nasa.gov/`

[6] `http://biopax.org/`

[7] `http://linguistics-ontology.org/`

[8] `http://wiki.dbpedia.org/`

describes the use of natural language text as a human interface to ontologies. The basic premise of this model is that the underlying logical structures in ontologies can be masked via natural language text during user interaction with the ontology, for example while editing or querying the ontology. Such an interface allows for proposing upcoming suggestions consistent with the existing knowledge in the ontology in a human friendly way. Based on this model, [Franconi *et al.*, Franconi *et al.*, 2010, 2011] present a natural language based query interface to ontologies and [Perez-Beltrachini *et al.*, 2014] extend it by allowing for incremental query generation. [Evans and Power, 2003], on the other hand, present a natural language interface for ontology authoring.

In sum, the development of the semantic web and the proliferation of Knowledge Bases call for many applications in which natural language generation can significantly aid human interactions. There are thus practical reasons to work on surface realisation from Knowledge Bases. Importantly however, this large set of homogeneous, logical, data is also a great opportunity for the development, evaluation and comparison of surface realisers. As is well known, the input to NLG can be varied (numerical, logical, linguistic) which makes such comparisons difficult across heterogeneous data formats. The current availability of large quantities of Knowledge Base data encoded in a uniform formalism (e.g., in RDF standard) makes such a comparison now possible. There are thus both practical and theoretical reasons to explore surface realisation from knowledge bases.

1.3 Research Issues

There are two main issues that need to be tackled when generating sentences from Knowledge Base data.

First, the mapping between data and text must be accounted for.

As shall be detailed in the next chapter, existing approaches to surface realisation address this requirement by using templates, grammars or a direct data-to-string mapping. Templates are partially complete linguistic expressions containing gaps (slots) which need to be filled up by the data coming from the input for a successful generation. A grammar specifies a set of transformation rules; each one mapping some portion of input data to corresponding syntactic constituents and the output text is generated by combination of those rules. Finally, direct mapping approaches learn a data-to-text mapping from parallel data-text corpora. The mapping learned is then used to generate new sentences from unseen test data. Several works based on templates, grammars (both manually defined and automatically learnt) and direct mapping models have been proposed in the literature and integrated with symbolic as well as with statistical techniques.

In this thesis, we provide new methods for addressing this issues in the supervised and weakly supervised settings. In the supervised setting, we automatically induce a grammar from a parallel data-text corpus and use it for surface realisation using an existing surface realiser where the choice of the best output is guided by a language model. In the weakly supervised setting, however, we use the supplied lexicon to extract a set of lexicalisations and subcategorisation frames for Knowledge Base symbols from non-parallel corpora and then use a probabilistic model to predict the best mapping of the syntactic arguments in a subcategorisation frame to the semantic arguments of the corresponding Knowledge Base symbol.

A second important issue that needs to be handled when generating text concerns the ranking of the alternative sentences generated by the generation system. Because of the paraphrasing power of natural language or because of the noise introduced by the generation system, a strategy must be defined for choosing the best paraphrase from among the many possible alternatives usually produced by the surface realisation of the input.

Using natural languages, humans can describe a given set of data in different contexts and via different expressions making use of lexical, phrasal and syntactic paraphrases. A successful surface realiser would mimick such "human-like" behaviour by producing coherent and varied sentences. This calls for techniques utilising resources (grammar and templates) that support alternative verbalisations and allow for fluency rating of sentences so produced. However, integral to this aspect is the drawback of over-generation that is, the production of overwhelmingly many sentences resulting from very generic templates or poorly constrained combination rules in the grammar.

By allowing the same input to be verbalised in different ways, the approaches we propose here allow for paraphrases. To choose the best output, we exploit a basic ranking technique using a language model in the case of the supervised approach and a general probabilistic model, in the case of the weakly supervised approach.

Overall, one distinguishing feature of the approaches proposed in this thesis, is that we make explicit use of linguistic knowledge to guide surface realisation. Many current approaches to data-to-text assume either the use of fixed templates or a direct data-to-text mapping (learned from a parallel corpus using some machine learning technique). In such approaches, simple linguistic constraints such as subject-verb agreement and more complex constraints such as the linking between syntactic and semantic arguments, are mostly ignored. In contrast, we propose approaches for surface realisation from Knowledge Base data which make explicit use of linguistic constraints. In the supervised setting, we induce from a parallel data-text corpus a Feature-Based Lexicalised Tree Adjoining Grammar which imposes strong constraints on the syntax-semantic interface i.e., on how the semantic arguments of a Knowledge Base relation and the syntactic arguments of a verb or a relational noun lexicalising that relation relate. Similarly, in the weakly supervised approach, we propose a probabilistic model which makes use of syntactic frames extracted from corpora and is designed to predict the linking between syntactic and semantic arguments. In short, in this thesis:

we argue for surface realisation approaches which combine explicit linguistic constraints with statistical learning either through the combination of an automatically extracted grammar with a surface realiser guided by a language model or through the use of a probabilistic model combined with a frame extractor.

All these issues specific to the surface realisation task justify for a stand-alone research; not necessarily in conjunction with the preceding phases (Content Planning and Micro Planning) for a full scale NLG system. Indeed, in recent years, there has been increasing interest in surface realisation. Thus, [Bohnet *et al.*, 2010] discuss surface realisation from the dependency trees of CoNLL-2009 shared task corpus [Hajič *et al.*, 2009]. The First Surface Realisation Shared Task [Belz *et al.*, 2011] was held in 2011 inviting research on surface realisation from dependency tree structures. More recently, the KBGen Challenge [Banik *et al.*, Banik *et al.*, 2012, 2013] was held in 2013 as a challenge on surface realisation from Knowledge Base data. Since then, many works have been inspired and focus on the surface realisation task alone – [Wang and Zhang, 2012], [Guo *et al.*, 2011], [Butler *et al.*, 2013], [Zarrieβ and Richardson, 2013], to name a few.

From the theoretical perspective, the aim of this thesis is to explore the issues raised by surface realisation from Knowledge Base data and to propose methods for addressing those issues in a linguistically principled way and with very minimal manual effort.

1.4 Contributions

In this thesis, we present two novel approaches to Surface Realisation from Knowledge bases.

Supervised Approach: The KBGen challenge [Banik *et al.*, 2012, Banik *et al.*, 2013] was designed to compare and evaluate surface realisation systems taking as input Knowledge Base data. Given a Knowledge Base fragment which forms a coherent unit, the task was to generate complex sentences which are both grammatical and fluent in English. The challenge made available to the participants a small (207 training examples) parallel corpus of text and KB fragment pairs as well as lexicons mapping KB symbols to words and phrases. In the first part of this thesis, we present a corpus-based method for inducing a Feature Based Lexicalized Tree Adjoining Grammar (FB-LTAG) from a parallel corpus of text and data. The resulting extracted TAG includes a unification based semantics and can be used by an existing surface realiser to generate sentences from KB data. We apply our induction method to the KBGen data, use an existing surface realiser and implement a ranking module to test the resulting grammar on KBGen test data. Experimental evaluation shows that our approach outperforms a data-driven generate-and-rank approach based on an automatically induced probabilistic grammar; and yields results that are close to those produced by a handcrafted symbolic approach. Moreover, a distinguishing feature of our approach is that it relies on an automatically extracted grammar that is compact (a few hundred trees) and linguistically principled (it follows the semantic and extended domain of locality principles of Tree Adjoining Grammar). We show that this feature allows for a hybrid approach where an automatically extracted grammar can be manually revised to improve both coverage and output quality.

Weakly Supervised Approach. A strong limitation of the supervised approach just described is that it requires the existence of a parallel corpus aligning a KB fragment with a sentence verbalising that fragment. In the second part of this thesis, we therefore explore an approach for surface realisation from KB data that uses a supplied lexicon but does not require a parallel corpus. Instead, we build a corpus from heterogeneous sources of text related to the domain of the Knowledge Base for which surface realisation is being developed (in this case, biology) and we use this corpus to identify possible lexicalisations of the KB symbols (classes and relations). We then use this corpus to estimate the probabilities of KB symbol lexicalisations, of subcategorisation frames and of the linking between the various syntactic and semantic arguments of a given event. We propose probabilistic models for the selection of appropriate frames and syntax/semantics mapping and use a scoring function that utilises the learnt probabilities to verbalise the given input. We present automatic and human based evaluations of the output sentences and analyze issues relevant to learning from non-parallel corpora.

In both these approaches, we use the KBGen data as a reference input. The KBGen data is itself derived from an existing biomedical ontology (namely, the AURA Knowledge base, [Chaudhri *et al.*, 2013]). Our methods are generic and can be easily adapted for input from other ontologies for which a parallel/non-parallel corpora exists.

1.5 Thesis Roadmap

The organisation of the chapters making up this thesis is as follows.

Chapter 2 provides a broad overview of NLG and studies the surface realisation task in detail. We discuss the relevant issues, the varying nature of inputs along with existing approaches to deal with them and put the study of surface realisation from ontologies into context.

In Chapter 3, we present a complete description of our supervised approach. In this chapter, we explore the surface realisation task from a grammar based approach. We learn a Feature based Lexicalised Tree Adjoining Grammar (FB-LTAG) with unification semantics from parallel corpora of Knowledge Base data and text. We present a novel method for inducing the grammar from corpora and use it for the surface realisation task. The grammar we induce is driven by the linguistic principles of TAG and takes into account both the syntactic and semantic information. We evaluate the output sentences using both the automatic and human ranking metrics and show that the grammar we extracted is conceptually simple, is adaptable to unseen test cases and restricts the overgeneration problem.

In Chapter 4, we describe the weakly supervised approach in detail. This chapter pursues the surface realisation task from a different perspective; i.e. the use of non-parallel corpora to learn verbalisation of event descriptions in ontologies. Here, we present a probabilistic approach which induces syntax/semantic mapping between the Knowledge Base data and surface text from a large domain corpora. We analyze the output sentences for their semantic/syntactic accuracy and identify future research avenues.

For both approaches, we depict the steps involved and we present an analysis of the problems faced.

Finally, in Chapter 5, we conclude by presenting a summary of our approaches and providing pointers for further research.

Chapter 2

Natural Language Generation and Surface Realisation

Contents

In this chapter, we present a broad study of Natural Language Generation (NLG) task and provide a comprehensive overview of the background materials relevant to our thesis. We introduce the NLG task, present high level architectures for systems implementing NLG and discuss in details the various issues relevant to NLG task. Then we focus on the study of the specific NLG issue we pursue in this thesis – the Surface Realisation task. We categorize the SR inputs in terms of the different data formats they use and present examples of each input type. Following this, we present a detailed discussion of the various approaches in SR and highlight the advantages and limitations of each approach. We then delve into the study of SR from the specific input format we adopt for our research work – the Knowledge Bases (ontologies). We present an introduction to ontologies and discuss existing works on veralisation of ontologies.

2.1 The NLG Task

Natural Language Generation can be defined as the task of generating natural language text to describe information encoded in machine representation systems (Eg: Database, KnowledgeBase, Logical formulas etc.). Information representation/storage in such systems are often in terms of complex relationship between data and are governed by formal constraints, making them difficult for immediate human consumption. It is thus very desirable that textual descriptions of information contained in these systems are communicated in a natural and fluent way to human users. NLG serves this purpose by generating textual descriptions of facts in some natural language such as English or Nepali.

2.1.1 Issues to Solve

Such a mapping from meaning representations to natural language implies several linguistically motivated transformations for a successful generation. In the following sections, we summarise the linguistic and pragmatic issues inherent to this task.

2.1.1.1 Content Planning

Content Planning is the task of identifying and structuring the relevant units of the input that can address the overall communicative goal of the generation system. It is a crucial issue for any NLG system as it determines the scope of the system on what it can express. Accordingly, content planning must be carried out in relation to the communicative goals which the particular NLG application needs to meet.

The Content Planning task can be divided into two subtasks. First is the Content Selection subtask which determines what part of the total input should be expressed in the output sentence. It involves identifying the content in the input which is necessary to build up a suitable response given the specified communicative goal of the NLG system. For example, given the communicative goal of defining a concept in an ontology, the content planning step could possibly select its superconcepts, subconcepts and siblings as relevant content units rather than its disjoint concepts. Second is the Content Structuring subtask in which the selected contents are ordered and organized into groups. It provides order to the sequence of information to be described in the generated text. To continue the example just discussed, a possible structuring scenario could be superconcepts followed by subconcepts followed by siblings; the siblings could further be grouped as a single unit of information and so on.

2.1.1.2 Micro Planning

The Micro Planning activity deals with the following issues :

- Lexicalisation : The content units identified from the content planning phase are still in their non-linguistic form. The lexicalisation task is concerned with identifying suitable lexemes (noun, verb, adjective inflections) for such content units which will be used to make up the natural language text describing the input. For example, in Sentence (1) verbalising the input from Figure 1.1, we used the lexemes "large" and "low" respectively to describe the numerical values above and below a certain threshold.

- Generating Referring Expressions (GRE) : In natural languages, different types of noun phrases (Pronoun, Proper Nouns, Definite descriptions, demonstrative NPs, Indefinite NPs, etc.) can be used to refer to an entity in the text. Such noun phrases, called referring expressions, must adequately identify each entity being talked about – both when the entity is first mentioned in the discourse (Initial Reference) or when the entity is subsequently referred to after it has been introduced once in the discourse (Subsequent Reference). GRE deals with the task of generating such referring expressions in the output text. For example, the use of a pronoun can introduce subsequent reference in multisentence text.

- Aggregation : Aggregation is the task of combining multiple content units into a common unit for linguistic expression. Typically, aggregation allows for the generation of ellided or coordinated structures. For example, the sentences "John ate an apple. The apple was rotten." can be aggregated using a participial construction as in the sentence "John ate an rotten apple.".

2.1.1.3 Surface Realisation

Surface Realisation is the task of producing surface text verbalising the information output. By deploying a direct mapping approach or using templates or a fully specified linguistic grammar, it deals with the task of producing linguistically correct and semantically precise verbalisations. As possibly many paraphrases can be realised from the same input, the surface realisation module should be able to generate all those while identifying the best for the final output. We defer the details to Section 2.2.

2.1.2 NLG Architecture

The NLG architecture is a conceptual model for the generation task. It provides a high level framework for addressing the above mentioned issues and their interactions. In most cases, the NLG architecture models the generation task as a sequential process (discussed in Section 2.1.2.1). However, other NLG systems propose that the issues are often an interplay and are expressed by a joint modeling system (discussed in Section 2.1.2.2).

2.1.2.1 Sequential Architecture

The NLG Reference architecture as proposed in [Reiter and Dale, 2000] has long been a widely used model for many NLG systems. The model is sketched in Figure 2.1 and demonstrates a pipeline structure for addressing the issues discussed in Section 2.1.1. Typically, given a communicative goal, the Document Planner first performs the Content Planning task. The output from this step is the Document Plan which is input to the MicroPlanner for resolving the issues with lexicalisation, GRE and aggregation. The Text Specification so obtained is usually a deep syntactic structure and is input to the Surface Realiser module to generate actual text in natural language.

2.1.2.2 Joint Architecture

The pipeline architecture presented above effectively implies that the modules are independent of each other and that each of the NLG issues can be handled in isolation. However, in practice, it is often the case that one affects the other – the boundaries

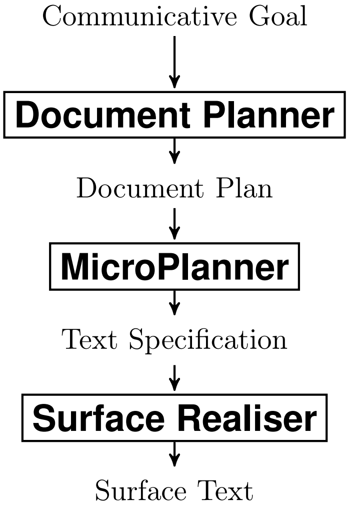


Figure 2.1: NLG Reference Architecture

of interaction between the modules disintegrate and mutually interact in a unified setting. For example, the lexicalisation of a content unit does vary depending upon the paraphrases produced by the surface realisation task. The defining aspect of a joint architecture is, therefore, that the modules make their decisions based on interrelated information.

[Danlos, 1989] present a study of several cases where such interrelated interactions come into play. For example, the verb conjugation varies depending upon the type of direct object used (in French, the auxilliary verb switches from *avoir* to *être* whenever the direct object changes from noun to reflexive pronoun : Jean **a** détesté Marie (John hated Mary) vs. Jean s'**est** détesté (John hated himself)), the surface structure of the sentence needs to be changed to address dative transformations (John gave [$_{NP}$ a book] [$_{PP}$ to Mary] vs. John gave [$_{NP}$ Mary] [$_{NP}$ a book]), the referring pronoun should adhere to pragmatic constraints (The pronoun "it" in the sentence "The dog jumped over the wall. It barked." can only refer to the dog and not the wall because the verb "bark" can only be used to describe animate actions) etc. All these examples justify the need for a joint modeling architecture. Several works implement joint architecture for generation – [Angeli *et al.*, 2010], [Dethlefs and Cuayáhuitl, 2012b], [Konstas and Lapata, 2012b], [Kondadadi *et al.*, 2013] etc.

2.1.3 NLG Inputs

NLG systems can target generation from various data types. Historically, NLG systems evolved from generating texts from database systems. Early applications such as the FOG system [Goldberg *et al.*, 1994] generated forecasts summaries from the weather database and SPOTLIGHT [Anand and Kahn, 1992] produced a textual description of market analysis from the retail sales database. NLG systems have also been built upon simple datasets of attribute-value pairs [Ratnaparkhi, 2000] and time-series numerical data [Belz, 2007] in the past.

Databases are still a major input type for NLG applications ([Angeli *et al.*, 2010], [Konstas and Lapata, 2012b]) but NLG systems targeting different data sources have also emerged – logical structures (as in [Lu and Ng, 2011], [Kondadadi *et al.*, 2013], [Cimiano *et al.*, 2013], [Ell and Harth, 2014]), discourse instructions (as in [Dethlefs and Cuayáhuitl, 2012b]) and linguistic structures (as in [Gali and Venkatapathy, 2009], [Bohnet *et al.*, 2010], [Guo *et al.*, 2008], [Wang and Zhang, 2012]).

2.2 Surface Realisation

We now focus our study on the SR task into which the main contribution of this thesis also lies. In sections below, we present a comprehensive study of existing methodologies and techniques in SR.

2.2.1 Inputs to SR

The input to the Surface Realisation task varies widely depending upon the NLG application at hand. For example, in NLG systems targeting verbalisation of ontology axioms, the input to SR consists of logical axioms (OWL expressions, RDF triples etc.) while in those that target verbalisation from databases, it involves the database records (Attribute-Value pairs, Concept Maps etc.). The syntax, structure and content of the input varies accordingly. Broadly, the input types to SR can be classified into the following three categories :

1. Data : This type of input comprises of collection of data, either as an unstructured collection of raw data or organized by their interrelationships into a set of records as in a database. The unstructured collection of data is also referred to as “flat” data. The time-series data in weather forecast generation systems, as described in [Reiter *et al.*, 2005], and [Belz, 2007] and collection of attribute value pairs, as described in [Ratnaparkhi, 2000] are examples of such input. Databases, on the other hand, provide structure to data by specifying their interrelationships. Such related sets of data, in the form of database records, have been used for generation in [Angeli *et al.*, 2010] and [Konstas and Lapata, 2012b].

2. Linguistic Structures : This kind of input comprises linguistically motivated syntactic/semantic representations. Examples include grammatical structures (such as the f-structures used in [Guo *et al.*, 2008] and dependency trees in [Wang and Zhang, 2012]) and discourse representations (as in [Kondadadi *et al.*, 2013] and [Dethlefs and Cuayáhuitl, 2012a]). Dependency trees from the CoNLL-2009 shared task [Hajič *et al.*, 2009] treebank have been used for input to SR in [Wang and Zhang, 2012] and [Bohnet *et al.*, 2010]. Also, [Basile and Bos, 2011] present a discourse representation corpus, SemBank, intended for NLG applications.

3. Logical Forms : Input of this type can come from various logical formalism. [Gerdemann and Hinrichs, 1990], for example, take first order predicate logic expressions as input. [Lu and Ng, 2011] take lambda expressions, [Androutsopoulos *et al.*, 2013] and [Stevens *et al.*, 2011] take OWL axioms, and [Cimiano *et al.*, 2013], [Ell and Harth, 2014] and [Duma and Klein, 2013] take RDF triples as input. In this thesis, we present our work on SR taking RDF triples from biomedical ontology as input.

2.2.2 Approaches to SR

Several appproaches to surface realisation have been proposed in the literature depending on the varying input type and the design requirements of NLG systems. Below, we outline these approaches and discuss in details the issues relevant to each of them.

2.2.2.1 Template Based Approaches

Template based approaches are based upon utilizing partially complete linguistic surface structures for generation. Such structures, called templates, contain gaps that should be filled with the appropriate content for a successful generation. Template based surface realisation can be illustrated with a simple example of verbalisation from a toy database. Figure 2.2a presents a small database of bus schedules. A template like the one shown in Figure 2.2b can be designed to generate natural language text describing the schedule of each bus. This template provides a partially complete text containing gaps that need to be filled with data (the corresponding variables supplying those data are shown in bold) from the input to generate sentences as shown in Figure 2.2c.

An early example of such template based approach is the mail-merge feature available in various document processing applications (such as in the MS-WORD

Bus No.	**Origin**	**Destination**	**Duration**
B100	Paris	Nancy	5 hours
C23	London	Paris	8 hours

(a) A toy database of bus schedules

The bus _____**(Bus No.)** departing from _____**(Origin)** reaches to _____**(Destination)** in _____**(Duration)**.

(b) Example template for describing bus schedules

Generated Sentences :
The bus B100 departing from Paris reaches to Nancy in 5 hours.
The bus C23 departing from London reaches to Paris in 8 hours.

(c) Example sentences generated using the template in 2.2b

Figure 2.2: Depiction of Template based Surface Realisation Approach

application).

Templates can be predesigned and handcrafted particular to the NLG system by considering the end requirements of the system. In such cases, considerable human time and expertise is needed both for determining the best verbalisation pattern and the content units from the input that can fill up the empty gaps in the template. [Van Deemter and Odijk, 1997], [McRoy *et al.*, 2003] and [Androutsopoulos *et al.*, 2013], for example, describe generation from predesigned templates.

An alternative approach to handwritten templates is extracting templates from domain related corpus. It involves learning templates for generation by observing the alignment between input data and surface text (Data-to-Text alignment) in the corpus.

Data-to-Text alignment is the process of matching input data to their surface text expression in sentences of a given corpus. It aims at finding exact or near-exact string match in the sentences of the corpus. The string matching can either target the input data itself (in case of linguistic input or whenever the sentences are annotated with the occurrences of input data) or attempt to match one of several different verbalisations possible for the data (in case of non-linguistic input). In the latter case, a lexicon mapping the input data to its several possible verbalisations is usually available with the input and the alignment involves matching one or several of such possibilities. For example, in [Duma and Klein, 2013] the input consists of natural language words which are aligned to corpus text via exact or near-exact string matching; in [Marciniak and Strube, 2005], the annotated sentences help direct alignment and in [Cimiano *et al.*, 2013] lexical entries for input are used in determining the alignment. Related works have also been described for non-linguistic input lacking lexical entries – [Liang *et al.*, 2009] propose a probabilistic method for text alignment of database values; [Trevisan, 2010] tokenize, pos-tag and use hand-written pattern rules to lexicalise ontology resources and [Walter *et al.*, 2013] propose a semi-automatic creation of lexicon for approximating the alignment of ontology concepts to words.

Additionally, the alignment can be targeted upon parallel or non-parallel corpora. In this thesis, we shall distinguish between alignment technique in parallel corpora (referring it as Supervised Approach) against that in non-parallel corpora but using a supplied lexicon (referring it as Weakly Supervised Approach). This distinction is important for two main reasons. First, the parallel corpora ensures that each instance of the input is verbalised by at least one sentence of the corpus and second, the sentence corresponding to an input instance completely verbalises all the data contained in that instance. Alignment from non-parallel corpora, on the other hand, doesn't guarantee these premises and is therefore prone to alignment failures.

Data-to-Text alignment technique forms an integral component of corpus based learning methods which target automatic learning of grammar/templates to represent the input by observing such correspondence in the training set examples. Crucially, the first step is the Data-To-Text Alignment which determines the match of data from the input to strings in the sentence (from a parallel or a non-parallel text corpora). Then, an appropriate grammar/templates is extracted by capturing the string of text between those alignments to represent the relationship expressed between the corresponding data of the input. This provides the training grammar/templates which is augmented via induction techniques for addressing the test data.

Corpus based learning offers a number of advantages over handcrafted or reusable grammar/templates approach – i) it is more robust, ii) has a larger coverage iii) is language and domain independent and iv) can express more flexibility in terms of linguistic expression compared to the rigid handwritten approaches. The downside is that it requires a big corpus and the high variety of output can sometimes overwhelm the capacity of the system. Addressing these limitations, the corpus (both parallel and non-parallel) being used for learning is ever growing in the NLP community and the high amount of output (i.e. overgeneration) is often constrained by some symbolic or statistical approaches particular to the application. To rank the extracted templates, [Lu and Ng, 2011] for example use a log-linear model, [Varges and Mellish, 2001] uses A* search based pruning and [Kondadadi *et al.*, 2013] an SVM model.

Several works describe corpus based templates learning for surface realisation using the Data-to-Text alignment technique. [Ratnaparkhi, 2000] align attribute values to word forms in the sentences and extract phrase templates. [Kan and McKeown, 2002] learn lexicalised phrase templates for semantic predicates from examples in annotated training corpus. [Kondadadi *et al.*, 2013] perform alignment of semantic predicates and entity names from DRS structure of corpus sentences to create templates. [Angeli *et al.*, 2010] extract templates to describe content from databases and [Duma and Klein, 2013], [Ell and Harth, 2014], [Cimiano *et al.*, 2013] present corpus based learning approach in extracting templates for verbalising RDF data. All these works apply supervised data-to-text alignment procedure with the exception of [Cimiano *et al.*, 2013] who use a non-parallel corpora crawled from the web.

2.2.2.2 Grammar-Based Approaches

In grammar-based approaches, a grammar is used to mediate the mapping between input data and surface text expression. A grammar consists of a set of rules, specifying the relation between portions of the input data, syntactic constituents and natural language expressions. For generation, the rules are combined as per the constraints imposed by the grammar and an output text is obtained. Thus, the grammar acts as an interface between the input data and the output surface text and models the syntactico-semantic interactions between them.

Surface realisation based on various grammar formalisms have been studied. Thus, [Rajkumar *et al.*, 2011], [White *et al.*, 2007] use Combinatory Categorical Grammar (CCG), [Carroll and Oepen, 2005] use Head-Driven Phrase Structure Grammar (HPSG), [DeVault *et al.*, 2008a], [Narayan and Gardent, 2012b] use Tree Adjoining Grammar (TAG) and [Zarrieß *et al.*, 2011] use Lexical Functional Grammar (LFG) based approaches.

Like templates, grammars used for surface realisation can be built in several ways. They can be predesigned and handcrafted manually specific to the NLG application at hand as is the case for instance, in the Functional Unification Grammar (FUG) discussed in [Elhadad, 1993]. In building such task-specific grammar, however, considerable human expertise and time is necessary and the grammar so developed eventually becomes limited to the particular application.

An alternative method is to reuse an existing wide-coverage grammar. Several wide coverage grammars have been developed in various formalisms and SR modules can benefit by simply reusing them, for free. [Copestake and Flickinger, 2000], for example, present the LinGO English Resource Grammar in HPSG formalism, [Hockenmaier and Steedman, 2007] present CCGbank in CCG formalism, [Doran *et al.*, 1994] present XTAG in TAG formalism and [King *et al.*, 2003] present PARC 700 DEPBANK in dependency grammar formalism. Surface realisation based on reusing such wide-coverage grammar have been discussed in [Carroll and Oepen, 2005], [Rajkumar *et al.*, 2011], [Cahill and Van Genabith, 2006], [Narayan and Gardent, 2012b] etc.

A downside of reusing an existing grammar is that it may be not be directly compatible with the input SR format and thus some format conversion becomes necessary. Indeed, such translation can be quite substantial and may demand as much effort as to handcraft the grammar particular to the application [Callaway, 2003]. [Busemann, 1996] outline the causes of such inadequacies – i) many wide-coverage grammars are designed in consideration to parsing as their primary application and thus adapting them for generation, which is seen as a reverse task of parsing, can be difficult [Russell *et al.*, 1990] ii) they may not sufficiently address needs specific to generation task such as presentation formats in tables, list etc and iii) they mostly model a syntactic structure and the semantic-syntactic mapping as needed for surface realisation may not be tightly integrated.

To circumvent such limitations, automatic grammar learning from corpus based examples have been proposed. As discussed earlier in Section 2.2.2.1, the Data-to-Text alignment comes into play in selecting text fragments representing data items of the input. Then, the text spans in between are extracted and encoded via rules in a chosen grammar framework. Several existing works describe this approach in the context of differing input types and using different grammars. For example, [Lu and Ng, 2011] learn an SCFG grammar to transform lambda expressions to surface text from the training examples. They align lambda sub-expressions to word sequences and learn constituent rules such that the training sentence is obtained when applying reduction operations (alpha and beta conversions) on the constituents. [Belz, 2007] accept collection of data values as input and present a semi-automatic approach in extracting CFG grammar for generation. They build up the terminal production rules automatically by aligning data to words chunks in the training sentences and manually author the non-terminal rules for combination of such word chunks. Finally, [DeVault *et al.*, 2008b] present a supervised corpus based learning of probabilistic Tree Adjoining Grammar by aligning frames (produced by a dialog system) with sentences in the training set. To our knowledge, none of the existing works describe grammar learning in a non-supervised setting.

2.2.2.3 Direct Mapping Approaches

An approach to surface realisation that is neither template based nor based on grammar is the direct mapping approach. Most of the existing works in this domain deal with linearizing unordered dependency tree to yield grammatical sentence with correct word order. [Bohnet *et al.*, 2010] present such an approach for surface realisation from the dependency trees of CoNLL-2009 shared task corpus [Hajič *et al.*, 2009] and [Wang and Zhang, 2012], [Ballesteros *et al.*, 2015] discuss surface realisation from the dependency trees of the 2011 Surface Realisation Shared Task corpus [Belz *et al.*, 2011]. Similarly, [Guo *et al.*, 2008] perform linearization upon Lexical Functional Grammar f-structures considering them as unordered dependency representation and [Filippova and Strube, 2007] linearize the dependency parse of sentences from Wikipedia for generation. Other notable works include those of [Ringger *et al.*, 2004] and [Zhong and Stent, 2009] where the authors present statistical models for ordering the unordered tree of syntactic constituents (in German and French) and positioning of adverbials in sentences (in English), respectively.

Since the inputs in such works already bear complete lexical information (or lack in few grammatical information such as functional nodes in [Ballesteros *et al.*, 2015] and dependency relations label in [Gali and Venkatapathy, 2009]), the major contribution in the direct mapping approach lies in finding the best ordering strategy. Accordingly, both stochastic and rule-based techniques have been proposed for determining the optimal word order in output sentences. [Gali and Venkatapathy, 2009] and [Guo *et al.*, 2008], for example, apply ngram filtering, [Bohnet *et al.*, 2010] and [Wang and Zhang, 2012] propose rule based linearization algorithm and [Ringger *et al.*, 2004] and [Zhong and Stent, 2009] use linguistic features (syntactic head information, semantic relations between words etc.) for building probabilistic decision models. The latter systems ([Ringger *et al.*, 2004], [Zhong and Stent, 2009]) which classify their word ordering decisions based on linguistic features are also referred to as classification-based SR and [Rajkumar and White, 2014] remark that there is not a clear distinction between such systems and the ones that work on linearising dependency tree structures since both of them perform incremental word ordering decisions over competing phrases rather than rank the final sequences of all word orderings possible.

Another line of work where a direct mapping is learned between input data and natural language can be found in the work of [Konstas and Lapata, 2012b] and [Wong and Mooney, 2007]. [Wong and Mooney, 2007] learn an SCFG grammar for mapping input (in variable-free tree representation such as CLANG) to surface text.

They then invert an existing semantic parser (WASP [Wong and Mooney, 2006]) for generation. Similarly, [Konstas and Lapata, 2012b] learn a Probabilistic Context Free Grammar (PCFG) from parallel data which captures the mapping between KB elements and natural language strings. Contrary to the grammar-based approach, in these cases, the grammar extracted from the data does not describe the syntax of natural language but directly, the relation between KB elements and natural language strings.

2.2.3 Discussion

Models that directly map the input to text (using templates as discussed in Section 2.2.2.1 or with direct mappings as discussed in Section 2.2.2.3) are short of linguistic knowledge that underlie the nature of human communication. In other words, such models are unaware of important linguistic constraints (both syntactic and semantic) that govern the well-formedness of utterances in any language. In Section 2.2.2.1, we presented templates which, in essence, simply fill up the gaps in the predetermined sentence structures with the data from the input. Either handwritten or data-driven (i.e. extracted from domain related corpus by observing the alignment between input data and sentences in the corpus), they do not support the modeling of syntactic/semantic relations among data present in the input. The same is true for the direct mapping approaches we discussed in Section 2.2.2.3. There, we presented the models for reordering of words in unordered dependency trees and extraction of rules for directly associating data in the input to strings as observed in the parallel texts. Both of there approaches do not accommodate linguistic information into them. In such approaches, simple linguistic constraints such as subject-verb agreement and more complex constraints such as linking between syntactic and semantic arguments are simply ignored.

In grammar-based approaches, however, the input data is mapped to linguistic constructs specifing their syntactic roles, such as the subcategorisation information and part-of-speech categories. The underlying grammar also specifies a set of linguistic rules which regulate the combinations of the syntactic constituents (selected by the input data) so as to produce linguistically correct verbalisation of the total input. In this manner, a grammar-based approach models the syntactic relation among data present in the input and allows for linguistically informed model of surface realisation. Further, it offers a number of advantages as compared to the other approaches.

First, grammars provide a principled approach for generalisation. For example, in a Lexicalised Tree Adjoining Grammar, the same grammar entry (tree) can be associated to different lexical items (such as *loves*, *hates*, *likes* etc.) realising the same syntactic function (transitive verb). This not only makes the grammar compact but also allows a generation system to make linguistically informed guess on mapping of unseen data (e.g. the relation *love*) to an appropriate grammar entry based on the mapping known for some other data (e.g. the relation *hate*) serving similar syntactic function. This is in contrast to the other approaches which are restricted to only what has been explicitly defined or observed in the training data.

Second, grammars present a modular approach to representation and composition of syntactic constituents in sentences, for example, by building separate grammar entries for core subcategorisation frames and optional modifiers but defining rules that allow them to compose, as needed, to form sentences of desired verbosity. Thus by adopting a grammar-based approach, one can map separate portions of input data to partial sentence structures in turn which can then be combined for generation. In general, one can expect that such mappings can be either be found for the entire input (thereby leading to a complete successful generation) or some portions of the input can't be mapped (thus leading to a partial generation). This is clearly an advantage over other approaches – templates, for example, are non-compositional in nature (although they can represent complex recursive structures, as noted by [Van Deemter *et al.*, 2005]) and can therefore be used only when they exactly cover the total input. By being able to build upon partial structures rather than attempting a complete matchup procedure, grammar-based approach can be said to be more robust form of surface realisation technique.

Finally, grammar-based approaches to surface realisation can benefit from a range of existing wide-coverage grammars. [Copestake and Flickinger, 2000], for example, present the LinGO English Resource Grammar in HPSG formalism, [Hockenmaier and Steedman, 2007] present CCGbank in CCG formalism, [Doran *et al.*, 1994] present XTAG in TAG formalism and [King *et al.*, 2003] present PARC 700 DEPBANK in dependency grammar formalism. Such grammars help the generation systems by providing wider coverage and eschewing the need of authoring task-specific grammars which are known to be manually intensive and time consuming. However, as discussed in Section 2.2.2.2, such wide-coverage grammars may not always be directly compatible with the input SR format at hand and therefore techniques for automatic grammar learning from corpus based examples have been proposed more recently.

An important practical issue concerning the grammar-based approaches to surface realisation is the choice of the best verbalisation from the set of multiple alternative sentences such systems can generate. Because the mappings (for the input) proposed by grammars reflect the expressive power of natural languages (paraphrasing, passivisation etc.), more than one paraphrases may be obtained for the same given input. Therefore, grading of the alternative verbalisations becomes necessary and this has led to the so called hybrid models of surface realisation whereby the generated sentences are ranked using some statistical or rule based criteria. Historically, the use of n-gram based ranking on the final set of output sentences to determine the best ouput has been widely reported for many existing works. However, as [Rajkumar and White, 2014] note, this approach is not very effective in dealing with sentences involving many constituents and many recent works incorporate syntatic knowledge from linguistic theory to obtain better ranking results. [Cahill *et al.*, 2007], for example, use log-linear model composed of features in the f-structure (in LFG formalism) and [Nakanishi *et al.*, 2005], [White and Rajkumar, 2009] use a combination of syntactic features (POS information, distance between head words, lexical entries of head words etc.) inspired from the works in HPSG parsing and CCG parsing, respectively to rank the output sentences. On the other hand, ranking decisions that operate on intermediate levels of realisation procedure (i.e. without first enumerating all the candidate generations) have been discussed in [Carroll and Oepen, 2005] (using conditional Maximum Entropy model built on features described in [Toutanova and Manning, 2002]), [DeVault *et al.*, 2008a] (using beam search which deploys weighted features learnt from training data), [Narayan and Gardent, 2012a] (by constricting the grammar search space using polarity filtering ([Gardent and Kow, 2005])) etc.

In this thesis, we propose two new grammar-based approaches to surface realisation. In Chapter 3, we present an automatic approach to extracting grammar from training examples (parallel data/text corpus) and use it for surface realisation of test inputs. Our approach is closest to [DeVault *et al.*, 2008b] in terms of extracting a grammar encoding syntax and semantics from training examples but enforces a tighter syntax/semantics linking information between the syntactic and semantic arguments . Further, we show that by using such syntax/semantics linking information as features on the grammar, we constrain the generation space and the ouputs are, in turn, ranked by a n-gram based language model. In Chapter 4, we present another method for grammar-based surface realisation; extracting a grammar from a non-parallel text corpora and focusing on learning of verbalisation patterns that respect the syntactic/semantic linking of arguments. The verbalisation patterns (syntactic frames) we use here are not templates; rather they describe the syntatic relations between an event and its arguments (subcategorisation information) and we use this information for learning of syntactic/semantic linking of arguments. We induce probabilistic models observing the verbalisation of semantic arguments (in the input) by syntactic roles in the frames and use them for ranking the n-best outputs.

2.3 NLG from Ontologies

An ontology is a conceptualisation of a given domain describing the entities, the classes (also called Concepts) and the relations that are present in that domain and axiomatising their behaviour. Typically, ontologies are encoded using a formal language such as OWL or RDF, they describe a hierarchical structure between concepts and model the relationship between via logical assertions (called axioms). [Gruber, 1993] define an ontology as "an explicit specification of a conceptualization", meaning that it serves as a formal model for expressing facts and relationship between facts. Note that an artifact modeling such theory in a computer representation is also referred by the same term. [Guarino *et al.*, 2009] differentiate between the use cases of this terminology – "Ontology" with uppercase initial and as an uncountable noun to refer to the philosohical discipline and "an ontology" with lowercase initial and as a countable noun to refer to a computational artifact. Pertaining to the computational artifact description of an ontology, say O, we reproduce below the formal definition presented in [Ehrig and Sure, 2004] :

$$O := (C, H_C, R_C, H_R, I, R_I, A)$$

"An ontology O consists of the following. The concepts C of the schema are arranged in a subsumption hierarchy H_C . Relations R_C exist between concepts. Relations (Properties) can also be arranged in a hierarchy H_R. Instances I of a specific concept are interconnected by property instances R_I . Additionally, one can define axioms A which can be used to infer knowledge from already existing one." [Ehrig and Sure, 2004]

Ontologies are used to build computational models of real world domain knowledge. In recent years, the use of ontologies for modeling domain knowledge has exploded in part due to the Semantic Web vision of exchanging and inferring from knowledge over the web. In this context, the Resource Description Framework (RDF) has been devised as a standard protocol for defining concepts and associations between them in the form of subject-predicate-object expressions (called triples). The subject and the object denote respective concepts and the predicate deontes a directed relation from the subject to the object concept. In this way, a set of RDF triples can model the domain knowledge in form of a directed multi-graph. Indeed, RDF is also the building block for higher level schema definition languages (such as RDFS and OWL) in the Semantic Web Stack.[9]

Although ontologies are well suited for computational representation and reasoning, they can be equally less intuitive and less insightful for a human user to grasp. This can be true for experts who are working on a broad scale ontology and need to analyze a fairly large chunk of already present logical statements in order to add/modify newer axioms to the knowledge base and also for beginners trying to get acquainted with the ontology formalism. Thus, in the course of design, maintenance and description of ontologies, a human user would perhaps like to seek necessary information from a given ontology in natural language. In such cases, it would be desirable to have access to a human comprehensible natural language description of the knowledge present in the ontology. This is where the motivation for NLG from ontologies lies in. Further, ontologies have been described to be well fitting for generation task. [Sun and Mellish, 2006] show that many ontology resources (concepts, properties and relations) widely use natural language words suitable for generation and [Power and Third, 2010] argue and provide evidence that current practices in ontology engineering tend to favour the generation task significantly.

Several works describe generation from ontologies. [Cimiano *et al.*, 2013] generate text from RDF data in an ontology modeling cooking recipes. [Ell and Harth, 2014] present a generic method for generation from RDF graph of concepts in any ontology and [Androutsopoulos *et al.*, 2013] present the NaturalOWL system, a generation system which verbalises complex logical axioms in ontology via multi-sentence text. These approaches will be discussed in more details in the following two chapters and we situate our appraoches with respect to them in the “Related Work” section of the respective chapters.

2.4 Conclusion

By now, we have discussed NLG and particularly the SR task in detail. We analysed the key issues in SR and discussed different approaches to SR from various input formats. We studied the advantages and limitations of each approach (template based, grammmar based and direct mapping) and presented a survey of the existing works. In the context of SR from KB data, we outlined the motivations for pursuing such goal, namely in terms of generating descriptions and aiding human interaction with ontologies.

In light of these discussions, this thesis proposes verbalisation of RDF triples

[9] `http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html`

in ontologies and presents two new grammar-based approaches for doing so. We present our approaches using a sample input derived from an existing biomedical ontology; however the approaches are generic and can be easily adapted to other ontologies. In contrast to the existing works on SR from ontologies, we develop corpus based learning methods (utilising Data-to-Text alignment techniques) and extract grammar resources for verbalising KB data both from the parallel (Supervised Approach, Chapter 3) and non-parallel text corpora (Weakly Supervised Approach, Chapter 4). For Data-to-Text alignment, we use a supplied lexicon (in Chapter 3) extending it with existing vocabularies (in Chapter 4) and to address the issue of high output variety, we utilize symbolic feature based constraints in TAG (in Chapter 3) and a probabilistic scoring approach (in Chapter 4). In the following chapters, we discuss our work in detail; compare and contrast them with existing works and highlight our innovative contributions.

Chapter 3

Verbalising Ontologies Triples – A Supervised Approach

Contents

This chapter is based on the following publications :

Bikash Gyawali and Claire Gardent. *LOR-KBGEN, A Hybrid Approach To Generating from the KBGen Knowledge-Base*, in Proceedings of the 14th European Workshop on Natural Language Generation (ENLG), pages 204 – 205, Sofia, Bulgaria, August 2013

Bikash Gyawali and Claire Gardent. *Surface Realisation from Knowledge-Base* in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), pages 424 – 434, Baltimore, USA, June 2014

This chapter presents a supervised approach to surface realisation from Knowledge Bases. We present a novel method for corpus based learning of Tree Adjoining Grammar (TAG) from training examples and discuss ways to generalise, expand and adapt it to cover test cases. The key feature of our approach is that grammar induction is driven by the extended domain of locality principle of TAG (Tree Adjoining Grammar) and takes into account both syntactic and semantic information. The resulting extracted TAGs include a unification based semantics and we use an existing surface realiser to generate sentences from Knowledge Base data.

3.1 Introduction

In the context of surface realisation from Knowledge Bases, handcrafted resources (either templates or grammar) have widely been used. Earlier works, for example, [Carenini *et al.*, 1994], [Paris, 1988], [Aguado *et al.*, 1998], [Galanis *et al.*, 2009] use handwritten templates for mapping semantic information in KBs to text. Other works, such as [Bontcheva and Wilks., 2004], [Williams and Power, 2010] and [Cimiano *et al.*, 2013] use a handwritten set of rules instead. As we discussed in Chapter 2, such handcrafted resources come at the cost of human intensive labor and become specific to the domain of the KB. Clearly, such limitations are major setbacks to the generation task for an ever growing domain of ontologies.

More recently parallel corpus based template learning methods have been reported for surface realisation, in particular to the RDF triples from an ontology in [Duma and Klein, 2013] and [Ell and Harth, 2014]. Again, as discussed in Chapter 2, the template extraction requires that the training sentences describe complete piece of semantic content in the input and when the test dataset bears new combinations of semantic content unseen during training, the non-compositional nature of templates implies that partial surface text observed from different training examples cannot be used to address semantic content combinations present in those examples.

We, therefore, explore an alternative, corpus-based grammar learning approach for surface realisation from ontologies in which we extract a symbolic compositional grammar from parallel data/text corpora. Given a training set consisting of pairs *(*$\{t_1, \ldots, t_n\}$*,*S*)* where $\{t_1, \ldots, t_n\}$ is a set of ontology triples and S is a sentence verbalising that set of triples, we develop a methodology for learning a TAG based grammar which mediates the mapping between KB triples and text. In addition, we automatically link the semantic content units from the input to the syntactic constructs in the extracted grammar thereby allowing for a tighter syntax/semantic integration. The set of grammar rules (TAG trees) are further augmented with unification-based semantics and therefore provide compositionality. Thus, our approach eschews both the handcrafting and non-compositionality problems discussed above. We follow a linguistically principled and conceptually simple approach and show that the induced grammar is compact; can be both expanded and adapted to cover for unseen cases and restricts the overgeneration problem.

In the following sections, we go into the details of our approach and discuss the relevant issues. We begin with a survey of related work in the domain of surface realisation from KBs in Section 3.2. In Section 3.3, we introduce the KBGen dataset upon which our experiments are based and in Section 3.4, we discuss FB-LTAG, the grammar formalism of choice for this work. In Section 3.5, we present a complete description of our approach, going into the details of the various subtasks involved (Data-to-Text Alignment in 3.5.1, various phases of Grammar Learning in 3.5.2, 3.5.4, 3.5.5 and 3.5.3 and Surface Realisation in 3.5.5). Then, we move on to present the results and their evaluation in Section 3.6. Section 3.7 discusses the problem cases, limitations and possible remedies and Section 3.8 concludes.

3.2 Related Work

The task of surface realisation from Knowledge Bases is related to work on concept to text generation.

Earlier work on concept to text generation mainly focuses on generation from logical forms using rule-based methods. [Wang, 1980] uses hand-written rules to generate sentences from an extended predicate logic formalism; [Shieber *et al.*, 1990] introduces a head-driven algorithm for generating from logical forms; [Kay, 1996] defines a chart based algorithm which enhances efficiency by minimising the number of semantically incomplete phrases being built; and [Shemtov, 1996] presents an extension of the chart based generation algorithm presented in [Kay, 1996] which supports the generation of multiple paraphrases from underspecified semantic input. In all these approaches, grammar and lexicon are developed manually and it is assumed that the lexicon associates semantic sub-formulae with natural language expressions. Our approach is similar to these approaches in that it assumes a grammar encoding a compositional semantics. It differs from them however in that, in our approach, grammar and syntax/semantic mapping information are automatically acquired from the data.

With the development of the semantic web and the proliferation of knowledge bases, generation from knowledge bases has attracted increased interest and so called ontology verbalisers have been proposed which support the generation of text from (parts of) knowledge bases. One main strand of work maps each axiom in the knowledge base to a clause. Thus the OWL verbaliser integrated in the Protégé tool [Kaljurand and Fuchs, 2007] provides a verbalisation of every axiom present in the ontology under consideration and [Wilcock, 2003] describes an ontology verbaliser using XML-based generation. As discussed in [Power and Third, 2010], one important limitation of these approaches is that they assume a simple deterministic mapping between knowledge representation languages and some controlled natural language (CNL). Specifically, the assumption is that each atomic term (individual, class, property) maps to a word and each axiom maps to a sentence. As a result, the verbalisation of larger ontology parts can produce very unnatural text such as, *Every cat is an animal. Every dog is an animal. Every horse is an animal. Every rabbit is an animal.* More generally, the CNL based approaches to ontology verbalisation generate clauses (one per axiom) rather than complex sentences and thus cannot adequately handle the verbalisation of more complex input such as the KBGen data where the KB input often requires the generation of a complex sentence rather than a sequence of base clauses.

To generate more complex output from KB data, several alternative approaches have been proposed.

The MIAKT project [Bontcheva and Wilks., 2004] and the ONTOGENERATION project [Aguado *et al.*, 1998] use symbolic NLG techniques to produce textual descriptions from some semantic information contained in a knowledge base. Both systems require some manual input (lexicons and domain schemas). More sophisticated NLG systems such as TAILOR [Paris, 1988], MIGRAINE [Carenini *et al.*, 1994], and STOP [Reiter *et al.*, 2003] offer tailored output based on user/patient models. While offering more flexibility and expressiveness, these systems are difficult to adapt by non-NLG experts because they require the user to understand the architecture of the NLG systems [Bontcheva and Wilks., 2004]. Similarly, the NaturalOWL system [Galanis *et al.*, 2009] has been proposed to generate fluent descriptions of museum exhibits from an OWL ontology. This approach however relies on extensive manual annotation of the input data.

The SWAT project has focused on producing descriptions of ontologies that are both coherent and efficient [Williams and Power, 2010]. For instance, instead of the above output, the SWAT system would generate the sentence: *The following are kinds of animals: cats, dogs, horses and rabbits.* . In this approach too however, the verbaliser output is strongly constrained by a simple Definite Clause Grammar covering simple clauses and sentences verbalising aggregation patterns such as the above. More generally, the sentences generated by ontology verbalisers cover a limited set of linguistic constructions; the grammar used is manually defined; and the mapping between semantics and strings is assumed to be deterministic (e.g., a verb maps to a relation and a noun to a concept). In contrast, we propose an approach where the grammar is acquired from the data and where the mapping between semantics and NL expressions is learned from the data rather than fixed a priori.

Recent work has considered data-driven generation from frames, lambda terms and data base entries.

[DeVault *et al.*, 2008b] describes an approach for generating from the frames produced by a dialog system. They induce a probabilistic Tree Adjoining Grammar from a training set aligning frames and sentences using the grammar induction technique of [Chiang, 2000] and use a beam search that uses weighted features learned from the training data to rank alternative expansions at each step.

[Lu and Ng, 2011] focuses on generating natural language sentences from logical form (i.e., lambda terms) using a synchronous context-free grammar. They introduce a novel synchronous context free grammar formalism for generating from lambda terms; induce such a synchronous grammar using a generative model; and extract the best output sentence from the generated forest using a log linear model. [Wong and Mooney, Lu *et al.*, 2007, 2009] focuses on generating from variable-free tree-structured representations such as the CLANG formal language used in the ROBOCUP competition and the database entries collected by [Liang *et al.*, 2009] for weather forecast generation and for the air travel domain (ATIS dataset) by [Dahl *et al.*, 1994]. [Wong and Mooney, 2007] uses synchronous grammars to transform a variable free tree structured meaning representation into sentences. [Lu *et al.*, 2009] uses a Conditional Random Field to generate from the same meaning representations.

Finally, more recent works propose approaches which perform both surface realisation and content selection. [Angeli *et al.*, 2010] proposes a log linear model which decomposes into a sequence of discriminative local decisions. The first classifier determines which records to mention; the second, which fields of these records to select; and the third, which words to use to verbalise the selected fields. [Kim and Mooney, 2010] uses a generative model for content selection and verbalises the selected input using WASP^{-1}, an existing generator. Finally, [Konstas and Lapata, Konstas and Lapata, 2012b, 2012a] develop a joint optimisation approach for content selection and surface realisation using a generic, domain independent probabilistic grammar which captures the structure of the database and the mapping from fields to strings. They intersect the grammar with a language model to improve fluency; use a weighted hypergraph to pack the derivations; and find the best derivation tree using Viterbi algorithm.

Our approach differs from the approaches which assume variable free tree structured representations [Wong and Mooney, Lu *et al.*, 2007, 2009] and data-based entries [Kim and Mooney, Konstas and Lapata, Konstas and Lapata, 2010, 2012b, 2012a] in that it handles graph-based, KB input and assumes a compositional semantics. It is closest to [DeVault *et al.*, 2008b] and [Lu and Ng, 2011] who extract a grammar encoding syntax and semantics from frames and lambda terms respectively. It differs from the former however in that it enforces a tighter syntax/semantics integration by requiring that the elementary trees of our extracted grammar encode the appropriate linking information. While [DeVault *et al.*, 2008b] extracts a TAG grammar associating each elementary tree with a semantics, we additionally require that these trees encode the appropriate linking between syntactic and semantic arguments thereby restricting the space of possible tree combinations and drastically reducing the search space. Although conceptually related to [Lu and Ng, 2011], our approach extracts a unification based grammar rather than one with lambda terms. The extraction process and the generation algorithms are also fundamentally different. We use a simple mainly symbolic approach whereas they use a generative approach for grammar induction and a discriminative approach for sentence generation.

3.3 The KBGen Dataset

We use the dataset provided by the KBGen challenge [Banik *et al.*, Banik *et al.*, 2012, 2013]. This challenge was designed to evaluate generation from knowledge bases. The KBGen dataset for training and testing generation systems was extracted from a biology knowledge base, namely the KB Bio 101 [Chaudhri *et al.*, 2013]. It was built by semi-automatically selecting content units from this knowledge-base in such a way that (i) the set of relations in each content unit forms a connected graph; (ii) each content unit can be verbalised by a single, possibly complex sentence which is grammatical and meaningful and (iii) the set of content units contain as many different relations and concepts of different semantic types (events, entities, properties etc.) as possible.

The foundational component of KB Bio 101 is the Component Library (CLIB) [Barker *et al.*, 2001], an upper ontology which is linguistically motivated and designed to support the representation of knowledge for automated reasoning [Gunning *et al.*, 2010]. CLIB adopts four simple top level distinctions: (1) entities (things that are); (2) events (things that happen); (3) relations (associations between things); and (4) properties (linking entities to their values). Accordingly, in the KBGen dataset, we have events which are concepts of type Event (e.g., Release-of-

Calcium, Bioethanol-Production, Isolation etc.), entities which are concepts of type Entity (e.g., Gated-Channel, Endoplasmic-Reticulum, Algae, Algal-Cell etc.), relations which are of type event-to-entity (e.g., Release-of-Calcium base Endoplasmic-Reticulum), event-to-event (e.g., Bioethanol-Production subevent Isolation), entity-to-event (e.g. Gated-Channel has-function Release-of-Calcium), entity-to-entity (e.g., Algae has-part Algal-Cell) and properties which are of type entity-to-value (e.g., Endoplasmic-Reticulum shape "hollow"). The KB Bio 101 subsets forming the KB-Gen dataset, therefore, encode knowledge about events, entities along with their properties and relations.

The KBGen dataset provides training and test data. Each item in the training data comprises some KB content (in the form of a set of RDF triples) paired with its corresponding verbalisation (a single sentence). In Figure 3.1 below, we present some examples (Figure 3.1a, 3.1c and 3.1b) of training items extracted from the KBGen dataset. The triples in the `:TRIPLES` section define the properties (if any) for the entity variables and the relations among the event and entity variables; the triples in the `:INSTANCE-TYPES` section define the predication true for the respective variables and the `:ROOT-TYPES` section provides the data types for the variables.

Following a practice introduced by [Angeli *et al.*, 2010], we use the term *scenario* to denote a KB content paired with its sentence. In the training input, there are 207 scenarios and the test input consists of 72 scenarios. We shall use the sentences in the training input to learn a grammar mediating the KB content to strings and use the sentences in the test input as reference sentences for evaluation of sentences generated for the KB contents in the test input.

To allow participants to focus on generation proper, the KBGen dataset also provides a lexicon listing words or phrases that can be used for verbalising entities and events present in the training and the test data. The lexicon maps event types to verbs, their inflected forms and nominalizations and each entity type to a noun and its plural form. For instance, the lexicon entries available for the entities and events present in Figure 3.1 is shown in Figure 3.2 below.

3.4 Feature-Based Lexicalised Tree Adjoining Grammar

In this section, we present an introduction to the Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG) with unification-based semantics since this is the formalism in which we will extract the grammar for our surface realisation task. A Tree Adjoining Grammar consists of a set of tree-shaped grammar rules which can be combined with each other via two tree combining operations, namely substitution

```
:TRIPLES (
   (|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
   (|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
   (|Gated-Channel64605|  |has-function| |Release-Of-Calcium646|)
   (|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
:INSTANCE-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Release-Of-Calcium|)
   (|Particle-In-Motion64582| |instance-of| |Particle-In-Motion|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Endoplasmic-Reticulum|)
   (|Gated-Channel64605| |instance-of| |Gated-Channel|))
:ROOT-TYPES (
   (|Release-Of-Calcium646| |instance-of| |Event|)
   (|Particle-In-Motion64582| |instance-of| |Entity|)
   (|Endoplasmic-Reticulum64603| |instance-of| |Entity|)
   (|Gated-Channel64605| |instance-of| |Entity|)))
```

Sentence :

The function of a gated channel is to release particles from the endoplasmic reticulum.

(a)

```
:TRIPLES (
   (|Diffusion-Of-Anion19310| |object| |Anion19306|)
   (|Diffusion-Of-Anion19310| |recipient| |Extra-Cellular-Matrix19307|)
   (|Diffusion-Of-Anion19310| |base| |Cell19296|)
   (|Diffusion-Of-Anion19310| |raw-material| |Membrane-Potential19309|)
   (|Membrane-Potential19309| |has-function| |Diffusion-Of-Anion19310|))
:INSTANCE-TYPES (
   (|Anion19306| |instance-of| |Anion|)
   (|Extra-Cellular-Matrix19307| |instance-of| |Extra-Cellular-Matrix|)
   (|Cell19296| |instance-of| |Cell|)
   (|Membrane-Potential19309| |instance-of| |Concentration-Gradient|)
   (|Diffusion-Of-Anion19310| |instance-of| |Diffusion-Of-Anion|))
:ROOT-TYPES (
   (|Diffusion-Of-Anion19310| |instance-of| |Event|)
   (|Anion19306| |instance-of| |Entity|)
   (|Extra-Cellular-Matrix19307| |instance-of| |Entity|)
   (|Cell19296| |instance-of| |Entity|)
   (|Membrane-Potential19309| |instance-of| |Entity|)))
```

Sentence :

Concentration gradient provides energy for the diffusion of anions from a cell to the extra cellular matrix.

(b)

```
:TRIPLES (
   (|Bioethanol-Production6816| |raw-material| |Carbon-Dioxide6823|)
   (|Bioethanol-Production6816| |result| |Ethyl-Alcohol6797|)
   (|Bioethanol-Production6816| |subevent| |Isolation6799|)
   (|Isolation6799| |object| |Starch6811|)
   (|Isolation6799| |site| |Laboratory6804|))
:INSTANCE-TYPES (
   (|Bioethanol-Production6816| |instance-of| |Bioethanol-Production|)
   (|Carbon-Dioxide6823| |instance-of| |Carbon-Dioxide|)
   (|Ethyl-Alcohol6797| |instance-of| |Ethyl-Alcohol|)
   (|Isolation6799| |instance-of| |Isolation|)
   (|Starch6811| |instance-of| |Starch|)
   (|Laboratory6804| |instance-of| |Laboratory|))
:ROOT-TYPES (
   (|Bioethanol-Production6816| |instance-of| |Event|)
   (|Carbon-Dioxide6823| |instance-of| |Entity|)
   (|Ethyl-Alcohol6797| |instance-of| |Entity|)
   (|Isolation6799| |instance-of| |Event|)
   (|Starch6811| |instance-of| |Entity|)
   (|Laboratory6804| |instance-of| |Entity|))
```

Sentence :

During the production of ethyl alcohol, which consumes carbon dioxide, starch is isolated in the laboratory.

(c)

Figure 3.1: Example Training Scenarios from KBGen dataset

`Release-Of-Calcium`	releases, release, released, release
`Particle-In-Motion`	molecule in motion, molecules in motion
`Endoplasmic-Reticulum`	endoplasmic reticulum, endoplasmic reticulum
`Gated-Channel`	gated channel, gated channels
`Bioethanol-Production`	produces, produce, produced, production
`Carbon-Dioxide`	carbon dioxide, carbon dioxide
`Ethyl-Alcohol`	ethyl alcohol, ethyl alcohols
`Isolation`	isolates, isolate, isolated, isolation
`Starch`	starch, starches
`Laboratory`	laboratory, laboratories
`Diffusion-Of-Anion`	diffuses, diffuse, diffused, diffusion
`Anion`	anion, anions
`Extra-Cellular-Matrix`	extra cellular matrix, extra cellular matrix
`Cell`	cell, cells
`Concentration-Gradient`	concentration gradient, concentration gradients

Figure 3.2: Example Entries from the KBGen Lexicon

and adjunction. TAG trees are of two types, initial and auxiliary trees. Initial trees represent non-recursive syntactic constituents such as noun phrases. In contrast, auxiliary trees represent recursive structures such as adjectives and adverbials.

In an initial tree, the internal nodes are non terminal syntactic categories and the leaf nodes are either terminals or non terminal syntactic categories marked for substitution (called substitution node and marked with a $\downarrow$ symbol). This is also true for the auxiliary trees but additionally, the auxiliary trees always bear exactly one leaf node, called the foot node, for adjunction operation. The foot node is marked with a $*$ symbol and is always of the same syntactic category of the root node of the tree. Both intial and auxiliary trees are called elementary trees and are said to be of type X whenever the root node of the tree has the syntactic category X.

Formally, a tree adjoining grammar is a tuple $(\sum, N, I, A, S)$ where $\sum$ is a finite set of terminals, N is a finite set of non-terminals with $(\sum \cap N = \emptyset)$; I is a finite set of initial trees; A is a finite set of auxiliary trees and S is a distinguished non-terminal symbol with $S \in N$.

In Figure 3.3 below, we present some examples of intial and auxiliary trees. Trees 3.3a (type NP), 3.3b (type NP), 3.3c (type S) and 3.3d (type VP) are examples of initial trees and the tree 3.3e (type VP) is an example of an auxiliary tree.

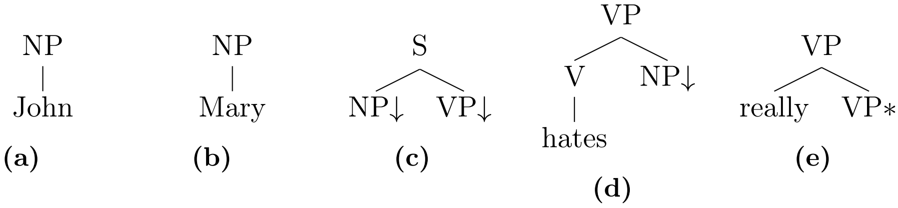


Figure 3.3: Example Initial and Auxiliary TAG trees

Lexicalised TAG (LTAG) is a variant of TAG in which all the trees in the grammar must have at least one leaf with a lexical item. The lexical item associated with the tree is also called the anchor of the tree and the motivation behind LTAG is that each TAG tree describes some lexical information with, when relevant, the subcategorization information of the anchor. In Figure 3.3 above, the tree 3.3c, does not bear any lexical items and therefore the grammar in Figure 3.3 is not in the LTAG formalism. An equivalent LTAG representation for the same grammar can be like the one shown in Figure 3.4 below.

The substitution operation is a simple procedure of replacing a substitution node in an initial tree with some other tree having the same syntactic category as the substitution node. The adjunction operation, on the other hand, inserts an auxiliary tree into an internal node of matching category in a given tree. Both operations

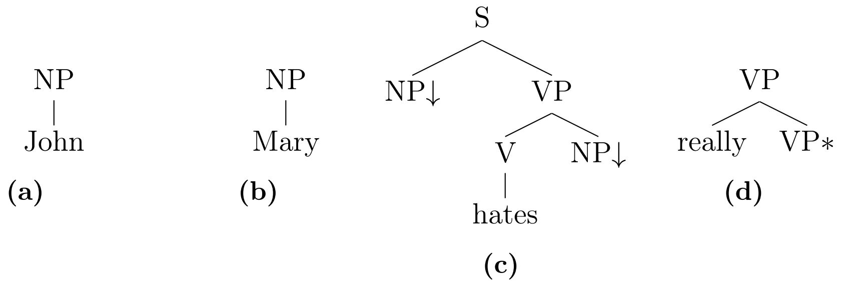


Figure 3.4: Example LTAG

produce new TAG trees, called derived trees.

Below, we show the substitution and adjunction operation in action. Figure 3.5 depicts the substitution operation. The substitution nodes in the tree 3.5b are replaced by the intial trees of matching NP type, trees 3.5a and 3.5c (shown via dotted arrows) and the derived tree in Figure 3.5d is obtained. The derived tree no longer bears any substitution nodes and is said to be complete. Figure 3.6 depicts the adjunction operation. During adjunction (shown via dotted arrows), the auxiliary 3.6b tree of type VP inserts into the internal node of matching type in the tree 3.6a derived tree in 3.6c is obtained. Note that the 3.6b tree can again adjoin with this derived tree to yield yet another derived tree 3.6d and so on.

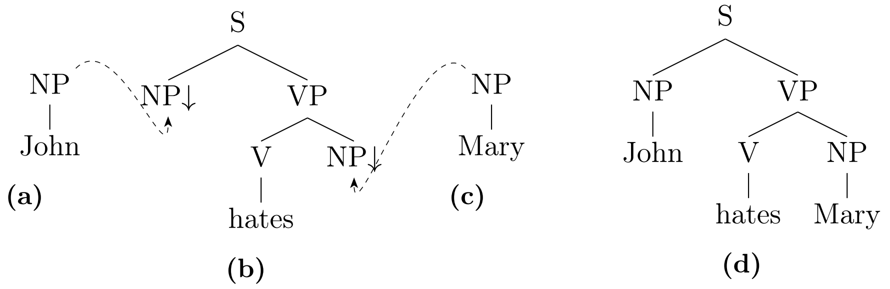


Figure 3.5: Substitution Operation in TAG

When parsing, the TAG trees whose lexical anchor match a word in the input string are retrieved from the grammar and repeatedly combined using the substitution and adjunction operations. All the derived trees that contain neither substitution nor foot nodes and cover the input string are retained. Thus, from the example grammmar in Figure 3.4, the following sentences can be parsed (among many others) :

(3) *John hates Mary.*

(4) *Mary hates John.*

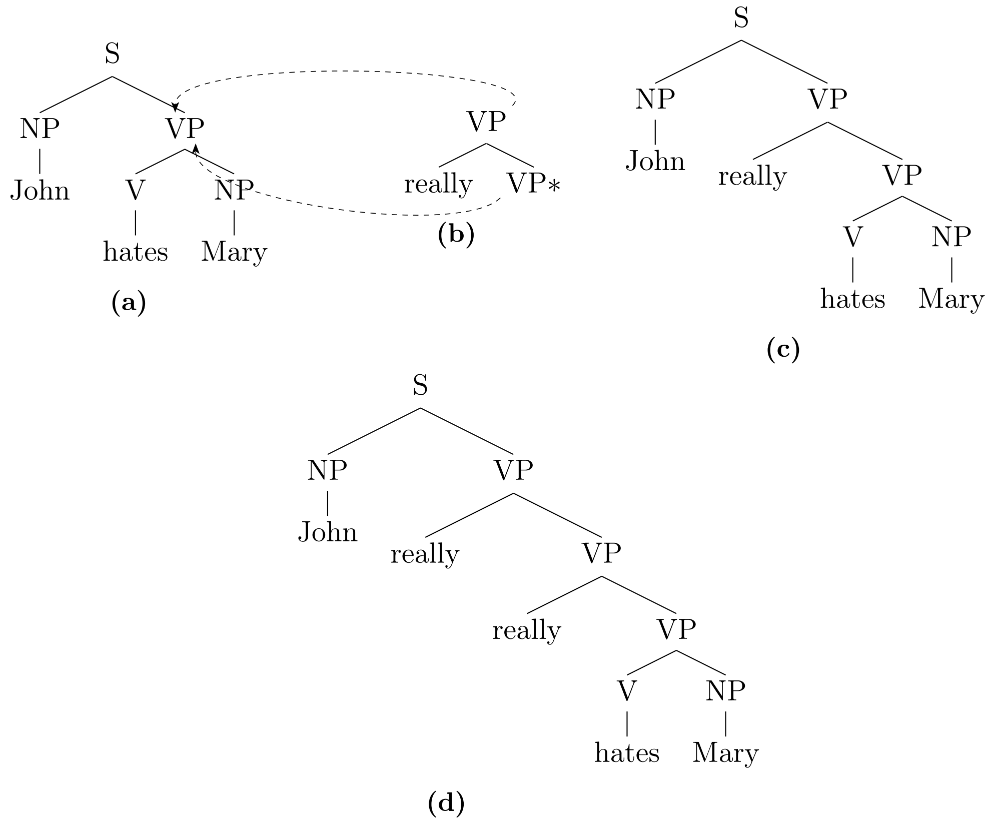


Figure 3.6: Adjunction Operation in TAG

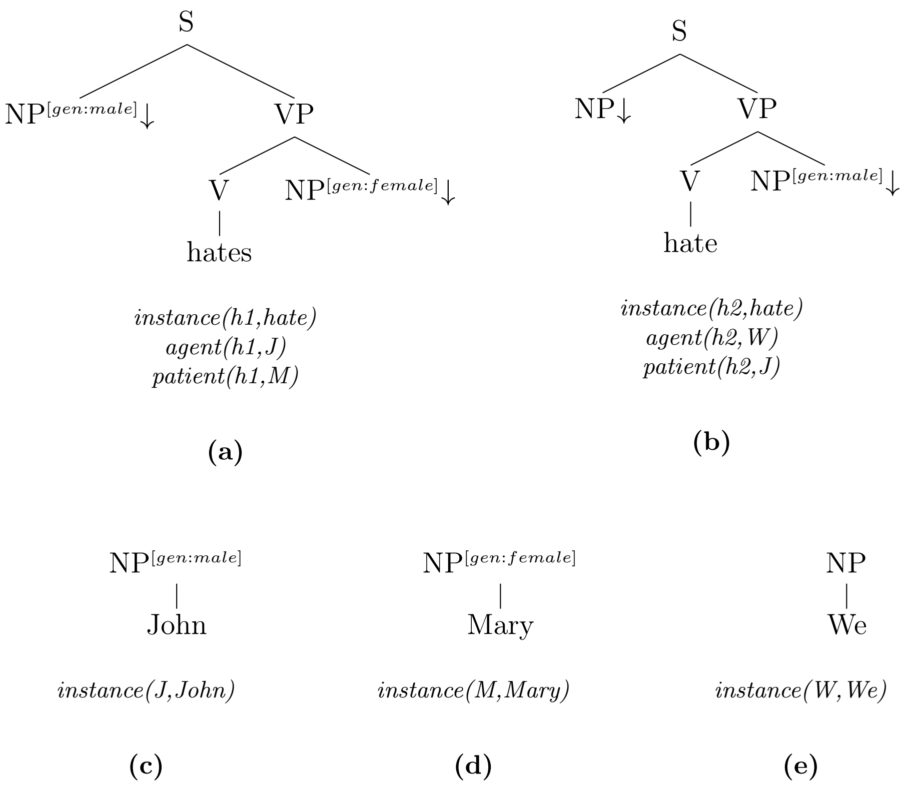
(5) *John really hates Mary.*

(6) *John really really hates Mary.*

(7) *Mary really really hates John.*

To generate sentences however, the grammar needs to relate words and syntax to semantic information. To this end, we use a Feature-Based Lexicalised Tree Adjoining Grammar with semantics where tree nodes are decorated with (non recursive) feature structures and each tree is associated with some semantics i.e., a set of literals whose arguments may be constants or variables. For generation, the trees in the grammar whose semantics subsume (part of) the input semantics are selected and combined via substitution and adjunction operations. The semantics of a derived tree is thus the union of the semantics of the all the trees contributing to its derivation. Figure 3.7 below shows an example of FB-LTAG equipped with semantics.

Feature unification during the substitution and adjunction works as follows. Substitution unifies the top feature structure of the substitution node with the top feature



(a) (b)

(c) (d) (e)

Figure 3.7: Example FB-LTAG with Semantics

structure of the root of the tree being substituted in. Adjunction unifies the top feature structure of the root of the tree being adjoined with the top feature structure of the node being adjoined to; and the bottom feature structure of the foot node of the auxiliary tree being adjoined with the bottom feature structure of the node being adjoined to. Figure 3.8 and 3.9 present the substitution and adjunction operation in FB-LTAG graphically.

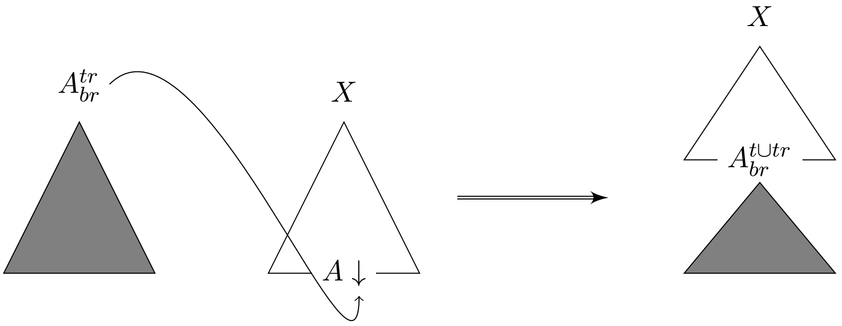


Figure 3.8: Feature Unification during Substitution in FB-LTAG

In FB-LTAG with unification-based semantics [Gardent and Kallmeyer, 2003],

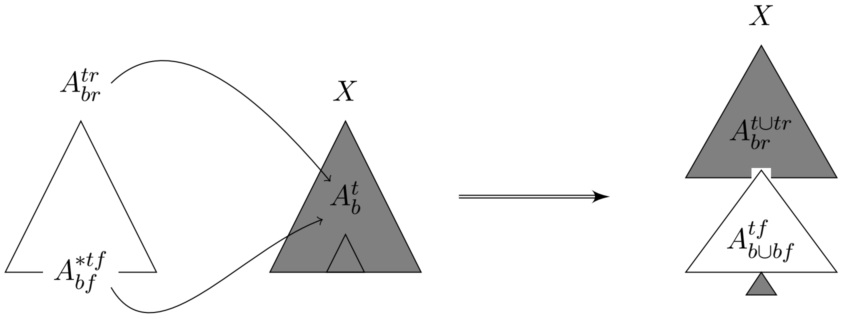


Figure 3.9: Feature Unification during Adjunction in FB-LTAG

the feature values in the nodes of the trees come from the variables in the semantic input instead. Importantly, the semantic variables are shared with syntactic variables (i.e., variables occurring in the feature structures decorating the tree nodes) so that when trees are combined as permitted by the semantic composition, the appropriate syntax/semantics linking is also enforced. For example, the nodes in the trees of the grammar shown in Figure 3.10 below are decorated with features comprising of semantic variables so as to obtain a FB-LTAG with unification semantics.

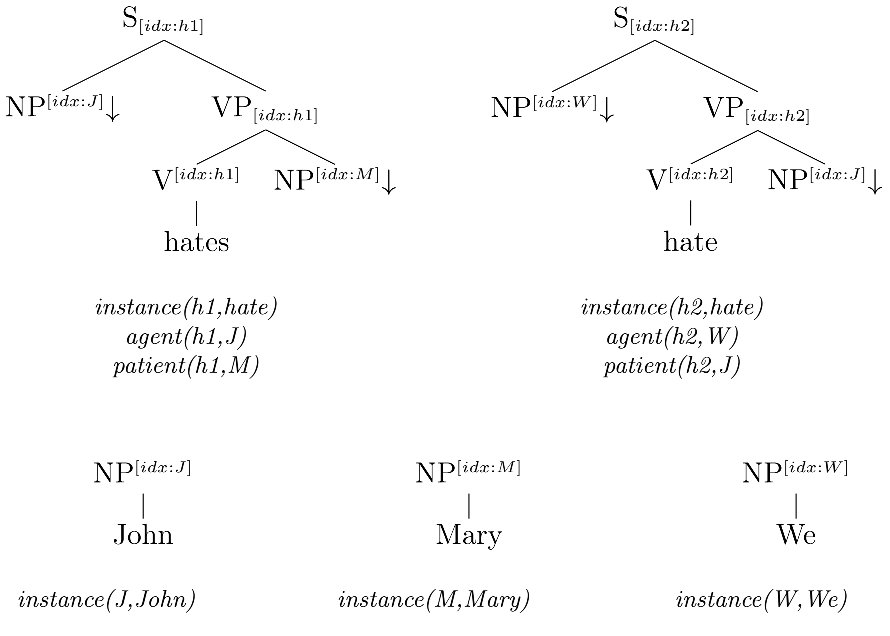


Figure 3.10: Toy FB-LTAG with Unification based Semantics

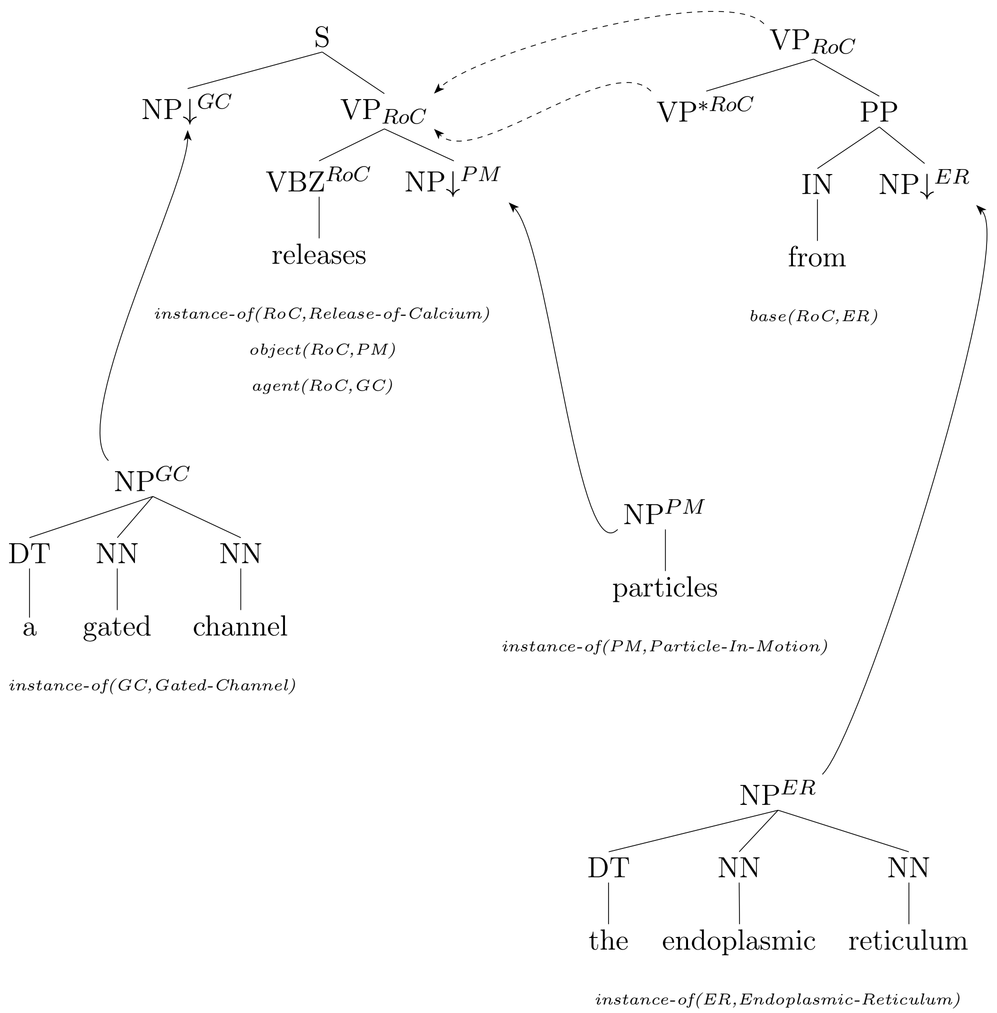
Now, if we have the following set of semantics relations as an input for generation :

instance(e,hate), *agent(e,X)*, *patient(e,Y)*, *instance(X,John)*, *instance(Y,Mary)*

the grammar in Figure 3.10 can only generate the sentence *John hates Mary* for this input; thanks to the sharing of variables in the semantics with the features in the trees. This example, although very small, demonstrates that such an approach constrains generation by encoding syntax/semantic linking constraints that restrict the set of generated sentences to those that conform with both the semantic input and the syntax/semantic linking encoded in the grammar. In contrast to the overgenerate-and-rank approaches which generate all possible derivations and filter the unwanted outputs later, here, the selection of trees suitable to the input task comes beforehand. In Figure 3.11 below, we present another example of generation scenario in the FB-LTAG formalism with unification based semantics for an actual input taken from the KBGen dataset. The solid lines indicate substitution operation and the dotted ones indicate the adjunction operation taking place among the trees. The variables decorating the tree nodes (e.g., GC) abbreviate feature structures of the form $[idx : V]$ where V is a unification variable shared with the semantics.

3.5 Approach

We target learning FB-LTAG trees with unification-based semantics from every example in the training set so as to build a grammar that can later be used for generation from the test set inputs. To that end, we process each training scenario in turn; and the task can be summarised as follows. From each of the training scenarios, we first align the KB data with the strings in the corresponding sentence using Data-to-Text Alignment (Section 3.5.1). Next, we induce FB-LTAG trees augmented with unification-based semantics from the aligned data (Section 3.5.2). To keep the grammar compact, we generalise the trees, grouping them into tree families and abstracting the trees from the lexical items they contain (Section 3.5.3). Then, we pursue an automated grammar expansion activity (Section 3.5.4) in which we look for possibilites of splitting the existing trees into smaller trees so as to have several smaller grammar units representing fewer semantic arguments of a relation at a time. Finally, we automatically adapt the trees obtained from the training scenarios to address the unseen cases in test set inputs (Section 3.5.5). The collection of all such trees constitute the grammar which is used with an existing surface realiser (Section 3.5.6) to generate from the test set inputs. Below, we describe each of these steps in detail.



Input Semantics :
instance-of(RoC,Release-Of-Calcium),
object(RoC,PM),agent(RoC,GC),base(RoC,ER),
instance-of(ER,Endoplasmic-Reticulum),
instance-of(GC,Gated-Channel),
instance-of(PM,Particle-In-Motion)

Generated Sentence :
A gated channel releases particles from the endoplasmic reticulum

Figure 3.11: FB-LTAG with Unification based Semantics for sample KBGen Scenario

3.5.1 Data-to-Text Alignment

Given a training scenario with the Sentence/KB pair (S, K), the alignment procedure associates each entity and event variable in K to a substring in S. To do this, we use the entity and the event lexicon provided by the KBGen dataset. For each entity and each event variables V in K, we retrieve their corresponding type (e.g., `Release-Of-Calcium` for the event variable `Release-Of-Calcium646` and `Gated-Channel` for the entity variable `Gated-Channel64605` in Figure 3.1a) as defined in the `:INSTANCE-TYPES` section of the input. Then we obtain all the phrases present for that type (e.g., *releases, release, released* for the type `Release-Of-Calcium` and *gated channel, gated channels* for the type `Gated-Channel` from the lexicon shown in Figure 3.2) from the KBGen lexicon and use a string matching operation to identify the word forms in the sentence S that match with one of these phrases. Once such a match is found, we align the corresponding variable V with the matching words in S. In Figure 3.12, we show an example alignment obtained for the sentence of the training scenario in Figure 3.1a using the lexicon in Figure 3.2. The bracketed notations depict the alignment; in each bracket, the matching words of the sentence are shown followed by the corresponding KB variable and separated by a comma. As we can see, the variables (`Gated-Channel64605`, `Release-Of-Calcium646` and `Endoplasmic-Reticulum64603`) are aligned by exact match of one of the phrases defined for their types in the lexicon with the words in the sentence.

However, there may not always be an exact match between the phrase associated with the type of the variable in the KBGen lexicon and the phrase occurring in the sentence. To account for this, we use some additional similarity based heuristics to identify the phrase in the sentence that is most likely to be associated with a variable lacking an exact match in the given sentence, i.e. for the entity variables, we search for nouns and for the event variables, we look for verbs in the sentence whose overlap with the variable type (or one of the lexical entries for this type in the lexicon) is not empty. For example, in Figure 3.1a, the entity variable `Particle-In-Motion64582` is aligned to the noun *particles* in the sentence based on the partial matching with its type `Particle-In-Motion`.

The function of a (gated channel, `Gated-Channel64605`) is to (release, `Release-Of-Calcium646`) (particles, `Particle-In-Motion64582`) from the (endoplasmic reticulum, `Endoplasmic-Reticulum64603`).

Figure 3.12: An example Data-to-Text Alignment

Note that this alignment procedure heavily relies on the input lexicon for matching of variables in the input to strings in sentences. For a parallel data/text corpus, it is only reasonable to expect that such a lexicon is provided. In those cases, the technique described here is fairly generic to establish the alignments of KB inputs even for different domains. For cases where such lexicon might not be available, alternative methods for learning such alignments from the corpus sentences themselves are worth exploring. [Liang *et al.*, 2009], for example, propose a generative model which learns the most probable word sequence in a sentence for the given data value in the input database. [Sun and Mellish, 2006] argue that the usage of (semi)linguistic terms to represent concepts and relations in ontologies is abundant and it is therefore feasible to derive lexical items from such observations; [Walter *et al.*, 2013] propose a semi-automatic method for creating lexical entries of concepts and relations present in ontologies and [Trevisan, 2010] tokenize, pos-tag and use hand-written pattern rules to lexicalise ontology resources.

3.5.2 Grammar Extraction

Grammar extraction from the training scenarios is carried out in following phases.

3.5.2.1 Variable Projection

To extract FB-LTAG trees from a training scenario, we first parse its sentence using the Stanford Constituency Parser[10] thereby generating a syntactic parse tree. From the Data-to-Text Alignment phase we know the substrings in the sentence that correspond to entity and event variables occurring in the input. Using this information, we then project the entity and event variables to the nodes in the syntactic parse tree to reflect headedness as follows. A variable aligned with a noun is projected to the first NP node up the syntactic tree or to the immediately dominating PP if it occurs in the subtree dominated by the leftmost daughter of that PP. A variable aligned with a verb is projected to the first VP node up the syntactic tree or in the case of a predicative sentence, to the root of that sentence. If the node projected with an event variable is immediately dominated by a parent node of the same category (e.g. a NP node projected with an event variable has an immediate parent node NP), we project the event variable to that parent node instead. We observed that these projection heuristics better represent the headedness information in the parse tree than that proposed by the Stanford parser. In Figure 3.13 below, we graphically present the variable projection procedure (via dotted paths) for the training scenario

[10] `http://nlp.stanford.edu/software/lex-parser.shtml`

given in Figure 3.1a. The variable names have been abbreviated[11] and are in bold.

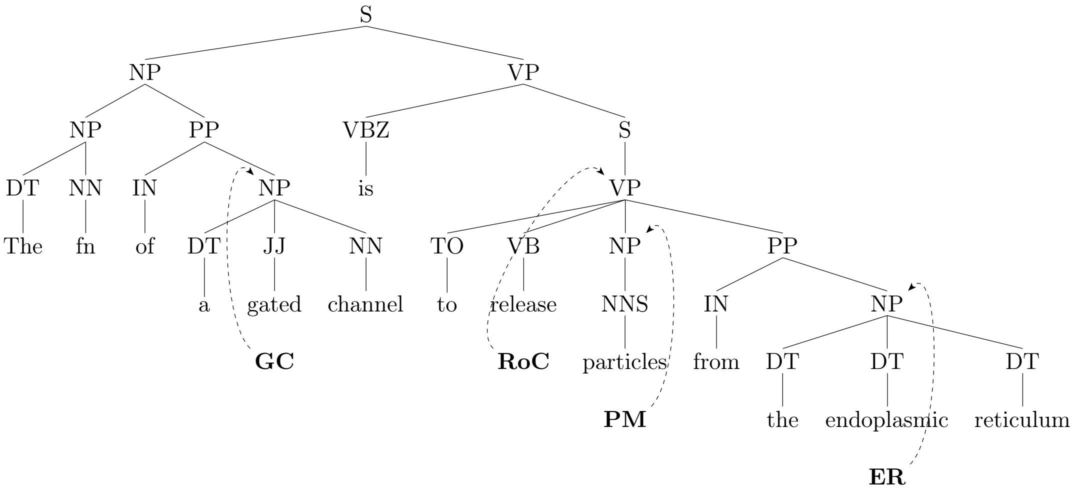


Figure 3.13: Visualisation of Variable Projection Procedure

Once entity and event variables have been projected up the parse trees, we extract FB-LTAG trees along with their semantics from the corresponding input scenario in several steps.

3.5.2.2 Entity Trees Extraction

In the first step, we extract trees representing the entity variables. From the parse tree of the sentence, we extract all the subtrees which are rooted at the nodes that have been projected with some entity variables. This results in a set of NP and PP trees and we associate each of these trees with the respective predication true of their labeling variables in the `:INSTANCE-TYPES` section of the input. Further, we assign the respective variable name as the top feature structure value for the root node of the extracted subtrees. Thus, from the parse tree in Figure 3.13, we obtain the FB-LTAG trees (with features sharing the semantic variables in the input) for the entity variables, *GC*, *PM* and *ER* as shown in Figure 3.14b, 3.14c and 3.14d below.

3.5.2.3 Relation Trees Extraction

Next, we extract trees representing the relations between variables as defined in the `:TRIPLES` section of the input. This involves the following three steps (**I**, **II** and **III**)

[11] For clarity reasons, we shall abbreviate the variable names in the trees throughout this text.

in sequence.

I. Event-to-Entity and Entity-to-Event Relations :

To capture trees representing relations between a given event and possibly many entities, we first identify all the entity variables in the input that are in relationship with the given event. That is, given an event variable X, we identify all its dependent entity variables m by looking for all relations R in the input that relate X to m in the fashion $(X\ R\ m)$. Lets refer to the set of all dependent entity variables identified for the event variable X as the set Y. Then, from the syntactic parse tree, we extract the minimal subtree containing the projection nodes for the event variable X and all and only the dependent entity variables of the event variable X. To this tree, we associate all the relations Φ in the input such that $\Phi = \{(Z \in Y) \wedge ((X\ R\ Z) \vee (Z\ R\ X))\}$. Using this procedure, we, therefore, extract (at once) a single tree representing all the event-to-entity and entity-to-event relations that exist in the input for the given event. In order to associate the extracted tree with unification based semantics, we assign the respective variable names as the top feature structure values for the nodes projected with the entity variables. Further, the node projected with the event variable X is assigned the bottom feature structure X and we mark the nodes projected with entity variables as substitution nodes. The tree in Figure 3.14a is an example of such tree extracted for representing all the relations existing between an event variable and all its dependent entity variables in the input. It captures all the relations of the type Event-to-Entity and Entity-to-Event existing between the event variable `Release-of-Calcium` and all its dependent entity variables `Particles-in-Motion`, `Endoplasmic-Reticulum`, and `Gated-Channel`, as present in the training scenario of Figure 3.1a.

II. Event-to-Event Relations :

For each relation (say R_1) between two events (say X_1 and X_2), the respective subtrees (say τ_{X_1} and τ_{X_1}) for those events have already been extracted from Step **I**. Now, to capture a tree expressing the relation R_1, we extract a minimal subtree (from the original parse tree) spanning the root nodes of τ_{X_1} and τ_{X_2} and assign to this subtree the relation R_1 as its semantics. Further, in this tree, we mark the root nodes of τ_{X_1} and τ_{X_1} as substitution nodes and decorate them with top structure values set to X_1 and X_2 respectively. The tree in Figure 3.16f is an example of such tree extracted for the event-to-event relation `subevent` in the training scenario of Figure 3.1c.

The training scenarios presented in Figure 3.1a, 3.1b and 3.1c are all examples of input bearing event-to-entity, entity-to-event and event-to-event relations. In Figure 3.14, 3.15 and 3.16, we show the complete set of trees extracted for these training scenarios respectively. The input triples (only the :TRIPLES section is shown) are repeated for convenience. Several cases are worth noting. As we can see from Figure 3.16, more than one subtree can be extracted while covering the relations present between variables in the same input – for example, Figure 3.16a and Figure 3.16g show two different trees obtained for representing the relations between event and entities and Figure 3.16f shows the tree obtained for representing the relation between event and event variables from the same input in Figure 3.1c. The entity variables are usually projected to the NP nodes and PP nodes and the event variables to VP (as in Figure 3.14a and 3.16a) or NP nodes (as in Figure 3.15a and 3.16g). Also note that the subtrees capturing relations between the variables may capture a verb (together with its arguments) occuring in a main clause (as in Figure 3.16a) along with other grammatical units such as prepositional phrases (as in Figure 3.14a and 3.15a) or in a relative or a subordinate clause (as in Figure 3.16g); thereby allowing for complex sentences.

```
:TRIPLES (
   (|Release-Of-Calcium646| |object| |Particle-In-Motion64582|)
   (|Release-Of-Calcium646| |base| |Endoplasmic-Reticulum64603|)
   (|Gated-Channel64605|  |has-function| |Release-Of-Calcium646|)
   (|Release-Of-Calcium646|  |agent| |Gated-Channel64605|))
```

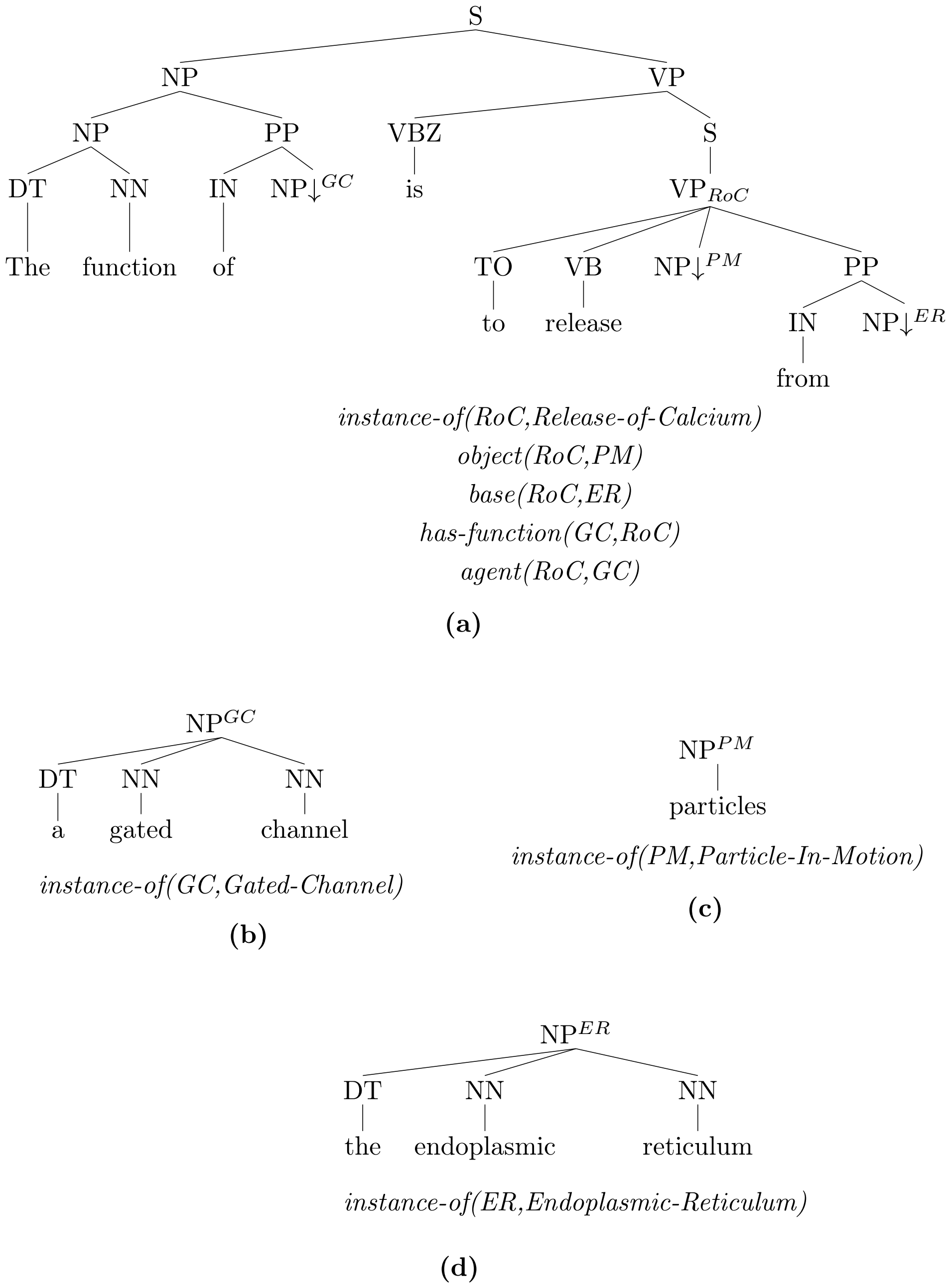


Figure 3.14: Grammar extracted for Training Scenario in Figure 3.1a

```
:TRIPLES (
   (|Diffusion-Of-Anion19310| |object| |Anion19306|)
   (|Diffusion-Of-Anion19310| |recipient| |Extra-Cellular-Matrix19307|)
   (|Diffusion-Of-Anion19310| |base| |Cell19296|)
   (|Diffusion-Of-Anion19310| |raw-material| |Membrane-Potential19309|)
   (|Membrane-Potential19309| |has-function| |Diffusion-Of-Anion19310|))
```

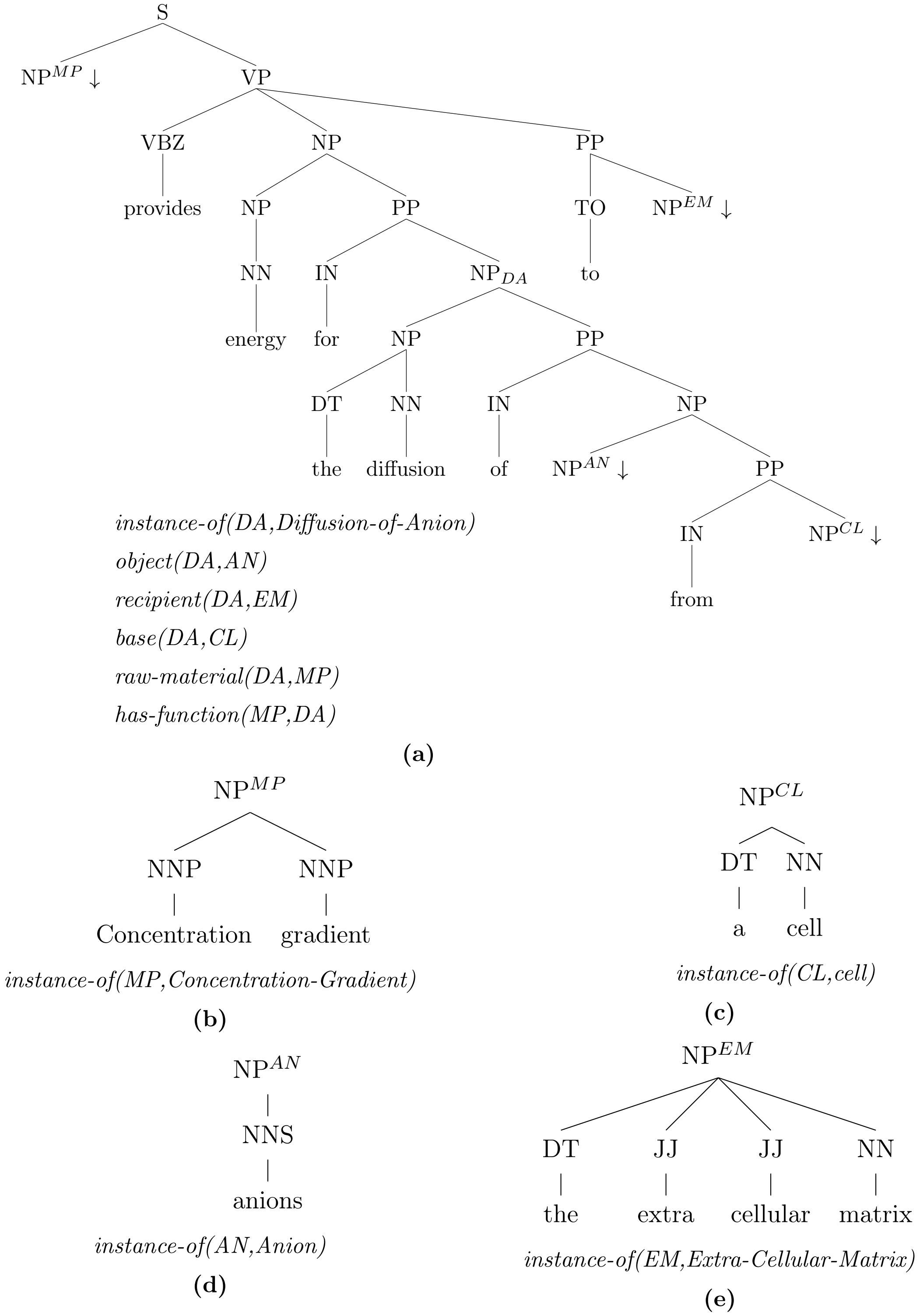
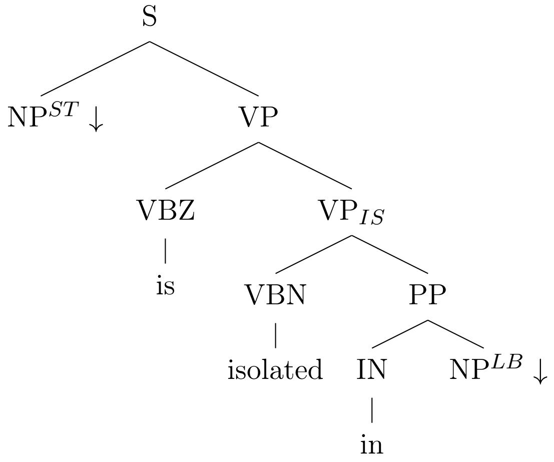


Figure 3.15: Grammar extracted for the Training Scenario in Figure 3.1b

```
:TRIPLES (
   (|Bioethanol-Production6816| |raw-material| |Carbon-Dioxide6823|)
   (|Bioethanol-Production6816| |result| |Ethyl-Alcohol6797|)
   (|Bioethanol-Production6816| |subevent| |Isolation6799|)
   (|Isolation6799| |object| |Starch6811|)
   (|Isolation6799| |site| |Laboratory6804|))
```

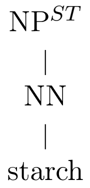


instance-of(IS,Isolation)

object(IS,ST)

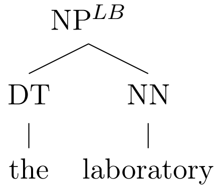
site(IS,LB)

(a)



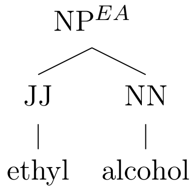
instance-of(ST,Starch)

(b)



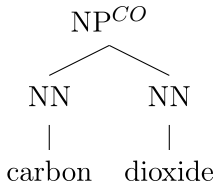
instance-of(LB,Laboratory)

(c)



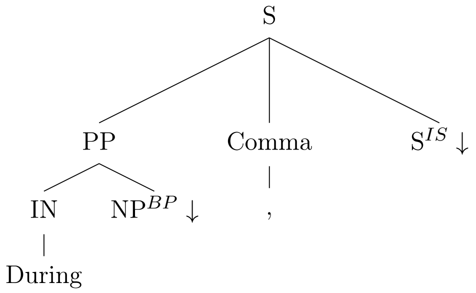
instance-of(EA,Ethyl-Alcohol)

(d)



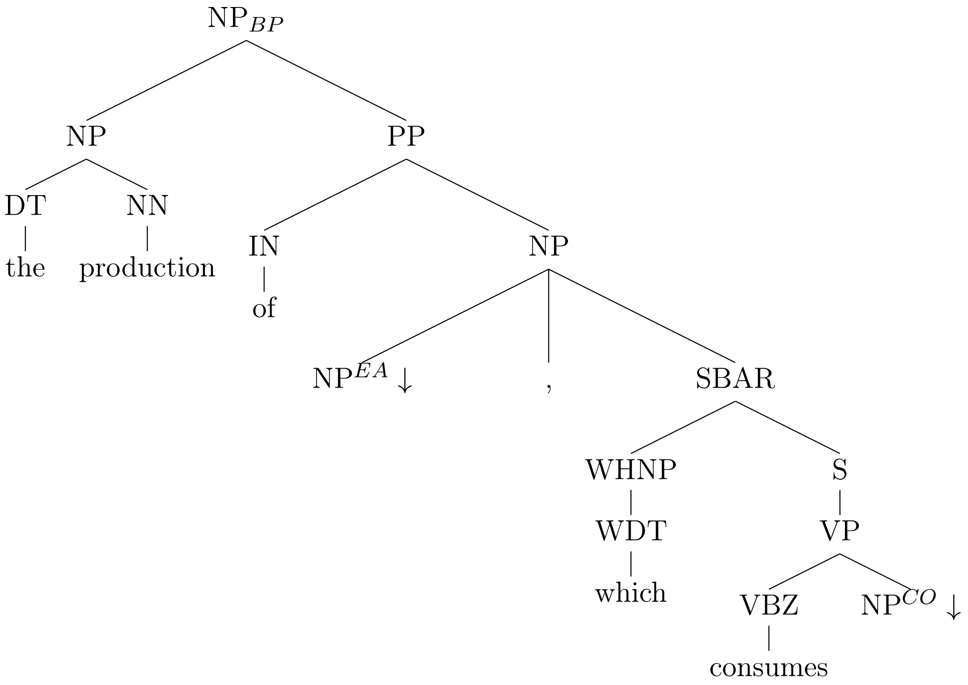
instance-of(CO,Carbon-Dioxide)

(e)



subevent(BP,IS)

(f)



instance-of(BP,Bioethanol-Production)
raw-material(BP,CO)
result(BP,EA)

(g)

Figure 3.16: Grammar extracted for the Training Scenario in Figure 3.1c

III. Entity-to-Entity Relations and Entity-to-Property Values :

The input can also bear relations between entities which are not bound to any event variables. Consider, for example, a different training scenario shown in Figure 3.17 below. This example is also set in the context of an input having property linking entity to its value and we shall use it to describe both these aspects.

The tree extraction capturing relation between entities takes place as follows. We first project the entity variables to the nodes in the parse tree (Figure 3.18a), extract their subtrees and associate them with the predication true of their respective variables (Figure 3.18d, 3.18e and 3.18f). Then, for each relation between entities, we extract a minimal subtree spanning the projection nodes of those entities and associate that subtree with the semantics of the relation (Figure 3.18b and 3.18c).

For entities linked by properties to their values (e.g. Tracheid linked by the property thickness to its value “thin” in Figure 3.18), the triples organization in the KBGen dataset is as follows. It involves two triple statements in the form $(X\ p\ Y)$

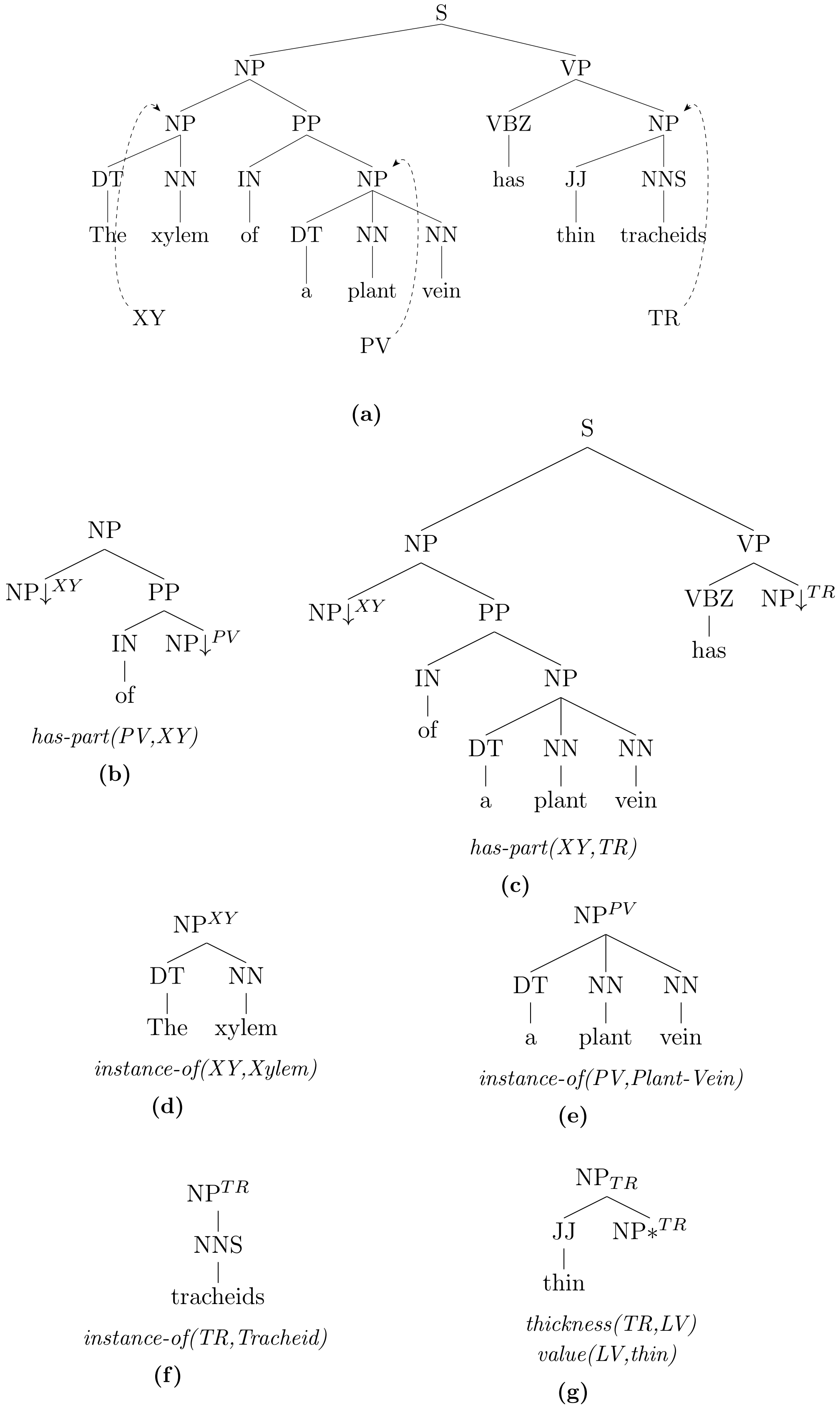
```
:TRIPLES (
   (|Plant-Vein3268| |has-part| |Xylem3269|)
   (|Xylem3269| |has-part| |Tracheid3271|)
   (|Tracheid3271| |thickness| |Length-Value3413|)
   (|Length-Value3413| |value| "thin"))
:INSTANCE-TYPES (
   (|Plant-Vein3268| |instance-of| |Plant-Vein|)
   (|Xylem3269| |instance-of| |Xylem|)
   (|Tracheid3271| |instance-of| |Tracheid|)
   (|Length-Value3413| |instance-of| |Length-Value|))
:ROOT-TYPES (
   (|Plant-Vein3268| |instance-of| |Entity|)
   (|Xylem3269| |instance-of| |Entity|)
   (|Tracheid3271| |instance-of| |Entity|)
   (|Length-Value3413| |instance-of| |Property-Value|))
```

Sentence :
The xylem of a plant vein has thin tracheids.

Figure 3.17: Training Scenario showing relation between entities with property values

and (Y *value* Z) where X is the entity variable (e.g. Tracheid), p (e.g. thickness) is the property linking that entity to an intermediate variable Y (e.g. Length-Value) which in turn relates to the actual value Z (e.g. "thin") via the predicate "value". From observations in the KBGen dataset, we see that such values are adjectives (e.g. "small", "rough", "thin" etc.) forming the JJ syntactic category under the NP head representing that entity in the parse tree (e.g. in Figure 3.18a, the value "thin" is an adjectival modifier of the entity tracheid). We therefore assume that the properties act as modifiers to the entities and construct auxialiary FB-LTAG trees of type NP to represent them. To such trees, we associate the semantics of both the triples (X p Y) and (Y *value* Z). The top and bottom feature values making up the foot and root node of the tree is the entity variable X and we lexicalise the tree with Z. The tree in Figure 3.18g is an example.

The resulting grammar from the collection of all trees extracted from all training scenarios is a Feature-Based Tree Adjoining Grammar with a unification-based compositional semantics as described in [Gardent and Kallmeyer, 2003]. In particular, our grammar differs from the traditional probabilistic Tree Adjoining Grammar extraction as described in e.g., [Chiang, 2000] in that it encodes both syntax and semantics rather than just syntax. It also differ from the semantic FB-TAG extracted by [DeVault *et al.*, 2008b] in that (i) it encodes the linking between syntactic and



(a)

has-part(PV,XY)

(b)

has-part(XY,TR)

(c)

instance-of(XY,Xylem)

(d)

instance-of(PV,Plant-Vein)

(e)

instance-of(TR,Tracheid)

(f)

thickness(TR,LV)
value(LV,thin)

(g)

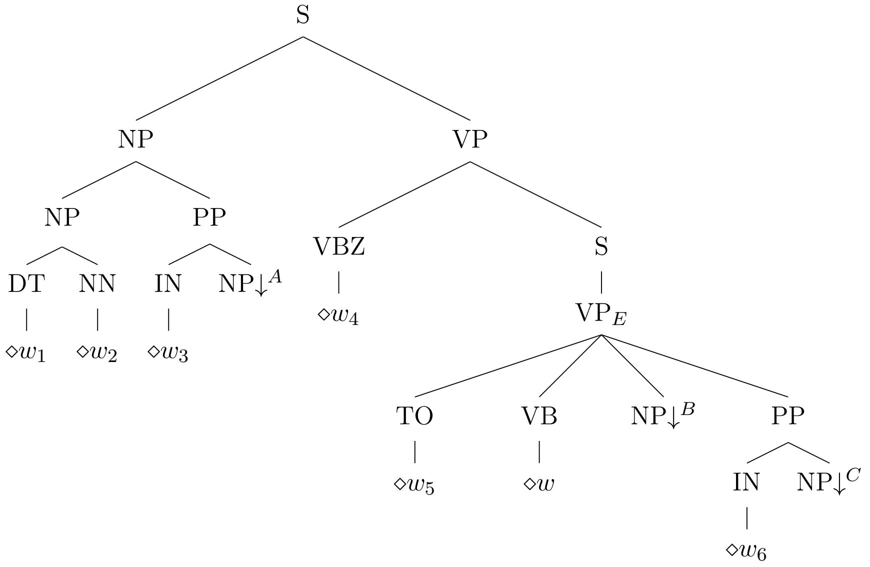
Figure 3.18: Grammar extracted for the Training Scenario in 3.17

semantic arguments; (ii) allows for elementary trees spanning discontiguous strings (e.g., *The function of X is to release Y*); and (iii) enforces the semantic principle underlying TAG namely that an elementary tree containing a syntactic functor also contains its syntactic arguments. Let us refer to the grammar we have from this stage as the *base* grammar; which we shall later expand (in Section 3.5.4) and adapt to unseen cases (in Section 3.5.5).

3.5.3 Grammar Generalisation

To keep the grammar size compact, we build tree schemas out of the trees in our grammar. A tree schema is obtained from a tree by replacing its terminal nodes, relation names and the semantic variables by generic variables whose values are instantiated at runtime (i.e. during the Surface Realisation phase only). To associate the terminal nodes, the relation names and semantic variables in the tree schema to their exact word forms, relations and input variables in the original tree, a separate lexicon is maintained establishing the correspondence. Figure 3.19a, for example shows the tree schema for the tree we presented in Figure 3.14a. The terminal nodes in this tree schema are generic variables, represented as w, $w_1 \ldots\ldots w_n$; the relation names are generalised to rel_1, rel_2, θ_1, $\theta_2 \ldots\ldots \theta_n$ (referring to the thematic roles of the event) and the semantic variables to A, B and C . The Lexicon in Figure 3.19b then holds the information mapping all these generic variables to their specific instantiations in the original tree. In the lexicon, the anchor w represents the word associated with the semantic variable for which this tree schema is generated and the coanchors $w_1 \ldots\ldots w_n$ denote rest of the terminal words in the tree. For each trees that we extract in our grammar, we create a new tree schema only when some existing schema (obtained from the trees in previously processed scenarios) doesn't suffice. In any case, a new lexical entry is to be created. Effectively, we have reduced the grammar size while retaining all the instances learnt from the entire training dataset.

The benefit of using tree schemas is twofold. First, it abstract away the trees from the word forms they contain and therefore we have a cleaner version of trees bearing structural information only. Second, and most importantly, it helps to keep the grammar compact by generalising over the set of similar trees we have extracted i.e. trees that bear the same syntactic/semantic structure but contain different word forms, relation names or input variables. For example, the same tree schema in Figure 3.19a can be used for the tree extracted for a different event (e.g. *carry* as verbalised in the sentence "*The function of a food vacuole is to carry solid substances using a pseudopodium.*") using a separate lexical entry shown in Figure 3.19c.



$rel_1(E)$, $\theta_1(E, A)$, $\theta_2(E, B)$, $\theta_3(E, C)$, $rel_2(A, E)$

(a)

Lexicon I

	Semantics	Anchor	CoAnchors
Ex.1	$\text{rel}_1 = \textsc{instance-of}$, $E = \textsc{release}$, $\theta_1 = \textsc{object}$, $B = PM$, $\theta_2 = \textsc{base}$, $C = ER$, $\theta_3 = \textsc{agent}$, $A = GC$, $\text{rel}_2 = \textsc{has-function}$	release	$w1$ → The $w2$ → function $w3$ → of $w4$ → is $w5$ → to $w6$ → from

(b)

Lexicon II

	Semantics	Anchor	CoAnchors
Ex.2	$\text{rel}_1 = \textsc{instance-of}$ $E = \textsc{carry}$, $\theta_1 = \textsc{object}$, $B = SS$, $\theta_2 = \textsc{instrument}$, $C = PD$, $\theta_3 = \textsc{agent}$, $A = FV$, $\text{rel}_2 = \textsc{has-function}$	carry	$w1$ → The $w2$ → function $w3$ → of $w4$ → is $w5$ → to $w6$ → using

(c)

Figure 3.19: Tree Schema to represent the tree in 3.14a and example entries for its lexicon

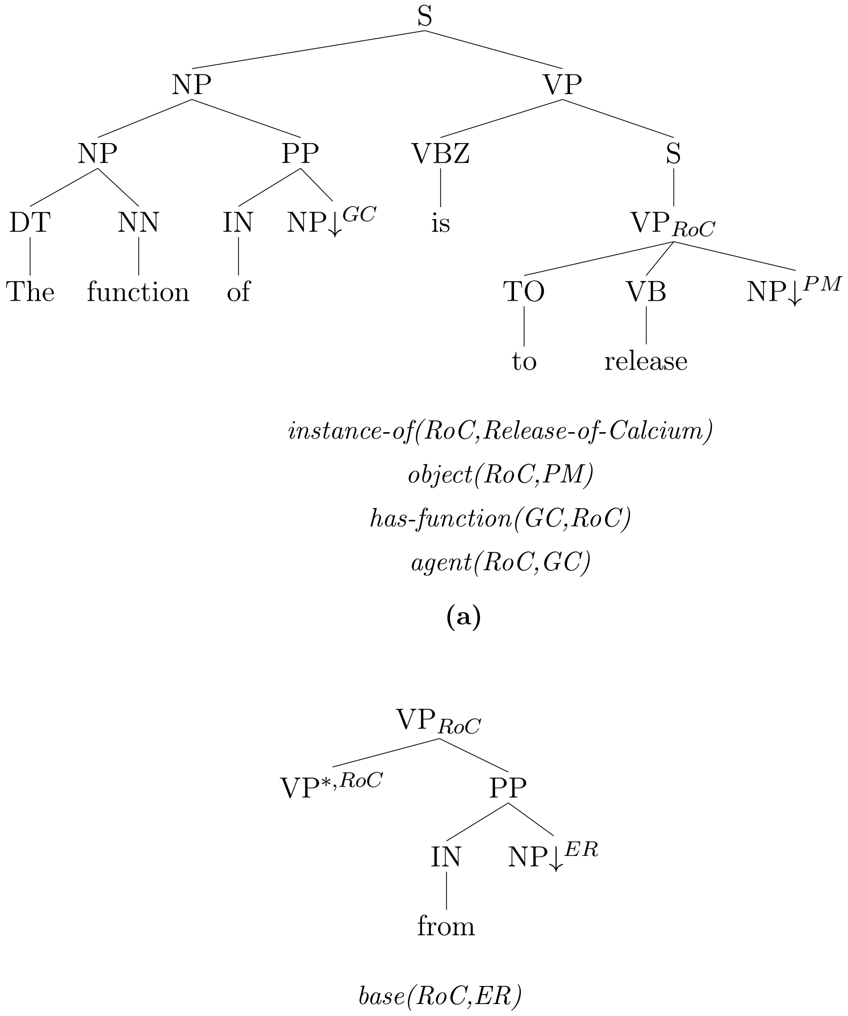
In our work, we follow the tree schema (along with the lexicon) based grammar representation, as just discussed. However, for sake of simplicity and intuitiveness, we shall restore to presenting complete trees in the sections to follow. In total, the grammar extracted from the training phase (i.e. after the completion of Grammar Expansion phase which we next discuss in Section 3.5.4) consists of 477 tree schemas. In Section 3.5.5, we present our approach for adapting the existing trees from the training phase to account for the test inputs. This will result in creation of new lexical entries for the test inputs but the count of tree schemas will effectively remain the same.

3.5.4 Grammar Expansion

The *base* grammar nicely captures the predicate/argument dependencies but it is very specific to the relation combinations seen in the training data. For example, from Figure 3.14a, we have a tree for an event variable in relation to several of its arguments but the tree is not usable to describe the same event in the test set when the event has lesser arguments (excludes optional modifiers, for example) or is augmented with new relations unseen in the training set. Thus, we look for an automated procedure that involves learning several smaller trees representing fewer relations at a time and adding them to the base grammar so as to obtain an *expanded* grammar.

We perform the grammar expansion as follows. From each tree τ in the *base* grammar which captures relations between multiple variables (i.e. X is the variable being described in relation to its dependents $D(X)$), we target the extraction of smaller subtrees that capture fewer arguments of X at a time. For each argument, $Y \in D(X)$ whose projected node is a leaf node and is immediately dominated by some PP node in the syntactic tree τ, we proceed to create a new auxiliary tree. The motivation here is that such PP nodes specify prepositional phrases which may act as modifiers to the variable X being described in the tree τ and are better represented via auxiliary trees in the TAG formalism.

For each argument in such configuration, we copy the subtree information of τ rooted at the PP node until the argument's projection node and construct a new auxiliary tree in the grammar. This newly constructed tree has the root and foot node with the same syntactic category as the node projected for the variable X in τ. Further, the bottom feature structure of the root node and the top feature structure of the foot node is assigned the value X and the semantic relation associated with the argument is assigned to this newly created tree. For instance, given the tree in Figure 3.14a, the PP tree associated to the argument Endoplasmic-Reticulum is extracted, marked as a auxiliary VP tree with features representing the event variable *Release-of-Calcium* and assigned with the relation *base(RoC,ET)*; thereby creating a new tree for the *expanded* grammar as illustrated in Figure 3.20b below.



instance-of(RoC,Release-of-Calcium)
object(RoC,PM)
has-function(GC,RoC)
agent(RoC,GC)

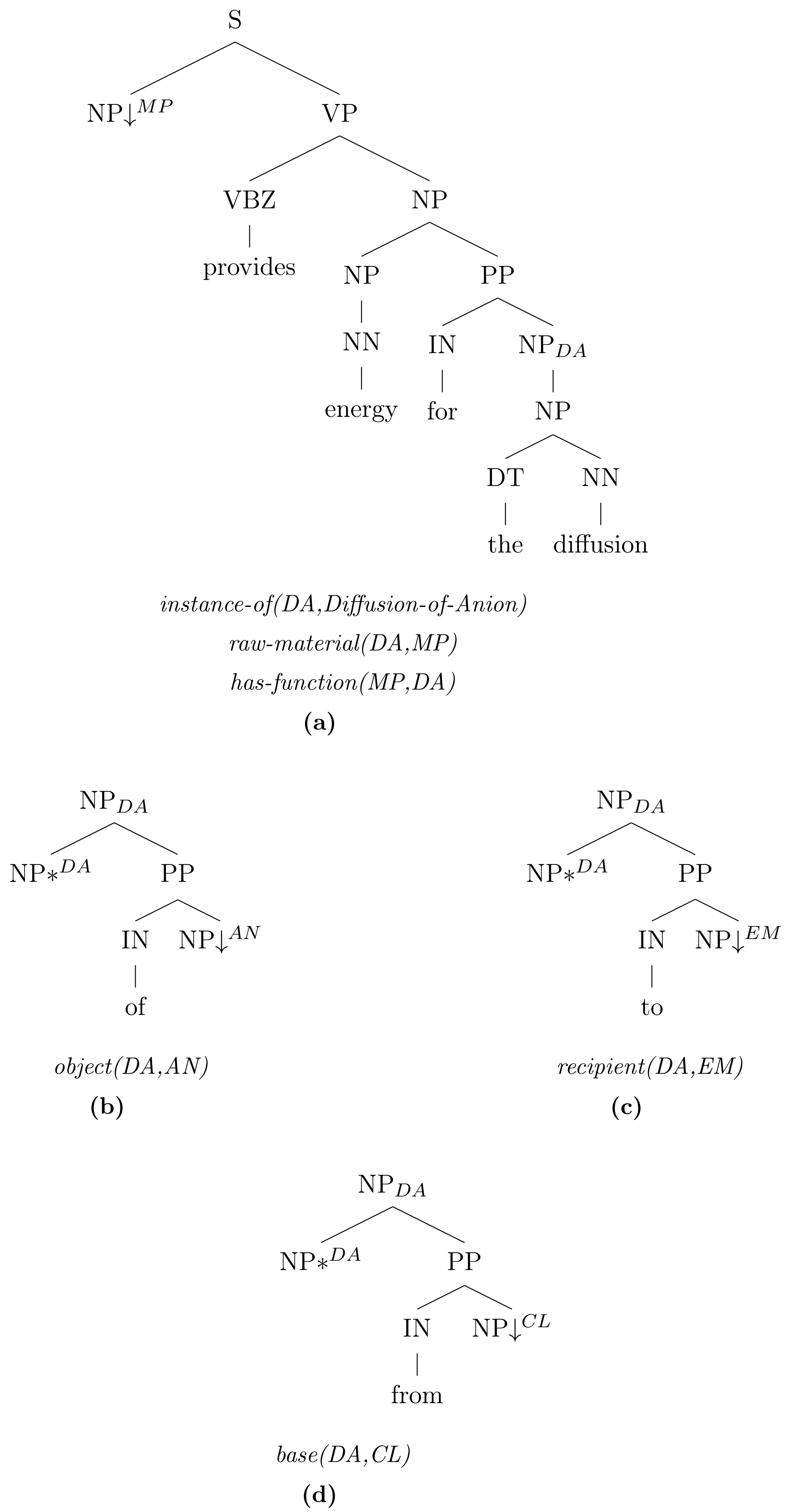
(a)

base(RoC,ER)

(b)

Figure 3.20: Trees Added by Grammar Expansion Activity on Figure 3.14

After all possible auxiliary trees from τ have been created in this manner, we construct one additional new tree τ' which is effectively the tree τ minus all the PP nodes from which the auxiliary trees were derived. The semantics associated with τ' is the semantics of τ minus the relations associated with the arguments present in the auxiliary trees created ealier. Figure 3.20a shows this tree created from the tree in Figure 3.14a. Note that this new tree represents the same event variable *Release-of-Calcium* but for a fewer number of arguments.



instance-of(DA,Diffusion-of-Anion)
raw-material(DA,MP)
has-function(MP,DA)

(a)

object(DA,AN)

(b)

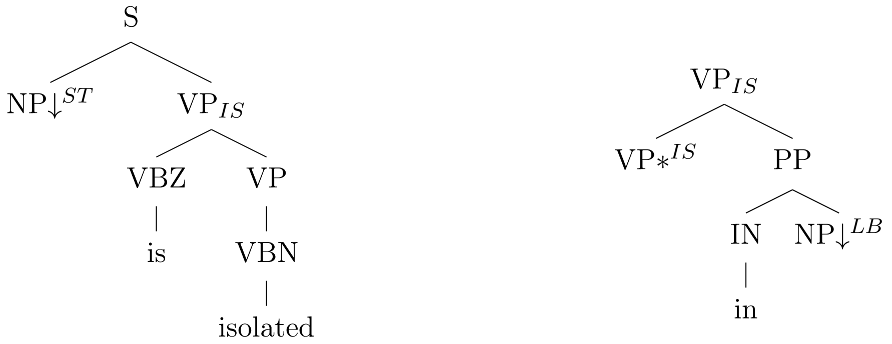
recipient(DA,EM)

(c)

base(DA,CL)

(d)

Figure 3.21: Trees Added by Grammar Expansion Activity on Figure 3.15



instance-of(IS,Isolation)
object(IS,ST)
(a)

site(IS,LB)
(b)

Figure 3.22: Trees Added by Grammar Expansion Activity on Figure 3.16

Figure 3.20a and 3.20b comprise the total trees obtained via grammar expansion from the trees in Figure 3.14 (corresponding to the Training Scenario 3.1a). In Figure 3.21, we show all the trees obtained after grammar expansion from trees in Figure 3.15 (corresponding to the Training Scenario 3.1b). The trees in Figure 3.21a, 3.21d, 3.21b and 3.21c are all obtained from the tree in Figure 3.15a. Similarly in Figure 3.22, we present the trees resulting via grammar expansion activity from the trees in 3.16 (corresponding to the Training Scenario 3.1c). There, both the trees in Figure 3.22a and 3.22b are obtained from the tree in Figure 3.16a.

The *expanded* grammar offers a number of advantages. First, it reduces overfitting by creating smaller, less specific, trees. Second, the new auxiliary trees provide independent verbalisation of semantic relations. This is in contrast to the *base* grammar where only a combination of relations could be verbalised all at once. Finally, it helps to account for test inputs which bear a combination of relations that were previously unseen in the trees extracted from any single training scenario but could be obtained via combination of smaller fragments of grammar arising from several distinct training scenarios. In the following section (Section 3.5.5), we present an example for such a case while the experimental results as discussed in Section 3.6 justify the overall advantages quantitatively.

At the completion of Grammar Expansion step, the *expanded* grammar consists of all the trees from the *base* grammar and all the newly constructed trees. The grammar learning from the training scenarios is said to be complete and we proceed to adapt this grammar to suit the test data requirements as discussed in next section.

3.5.5 Grammar Adaptation

Given the limited size of the training data, it is often the case that test input bears new entity and event variables unseen in the training data. However, our grammar extraction is modeled on capturing relations between variables and we can, therefore, take advantage of this feature to adapt the grammar for test inputs. We perform the grammar adaptation as follows.

For each unseen entity variable X defined by a predicate `X instance-of Pred_X`, we create a default NP tree with semantics of the form *instance-of(X,Pred_X)* and lexicalise it with the word form available in the lexicon. The root node NP is assigned a top feature structure with the value X.

For unseen event variables, we search for trees in our grammar that describe a similar event. A similar event is defined as an event which is associated with exactly the same set of event-to-entity and entity-to-event relations. The arguments themselves can differ but the cardinality of relations and arguments must be respectively equal. In other words, we are looking for an existing tree in the grammar which describes an event with similar semantics as the unseen event in the test input. When one is found, we copy the syntactic tree and associate it with the semantics of the unseen event. In addition, the word form verbalising the event variable in the tree is replaced by the word form provided for this unseen event variable in the lexicon and the variables making up the feature structures in the tree are updated to reflect the variables occuring in the relations making up the semantics of this unseen event.

```
:TRIPLES (
   (|Transfer22328| |object| |Electron22334|)
   (|Transfer22328| |recipient| |NAD-Plus22340|)
   (|Transfer22328| |agent| |Electron-Shuttle-System22339|)
   (|Electron-Shuttle-System22339| |has-function| |Transfer22328|))
:INSTANCE-TYPES (
   (|Transfer22328| |instance-of| |Transfer|)
   (|Electron22334| |instance-of| |Electron|)
   (|NAD-Plus22340| |instance-of| |NAD-Plus|)
   (|Electron-Shuttle-System22339| |instance-of| |Electron-Shuttle-System|))
:ROOT-TYPES (
   (|Transfer22328| |instance-of| |Event|)
   (|Electron22334| |instance-of| |Entity|)
   (|NAD-Plus22340| |instance-of| |Entity|)
   (|Electron-Shuttle-System22339| |instance-of| |Entity|)))
```

Figure 3.23: An Example Test Scenario

Lets consider a Test Scenario as shown in Figure 3.23 above. Figure 3.24 below shows the resulting grammar obtained by adapting our training grammar for this test scenario. Given the grammar we have induced from the three Training Scenarios in Figure 3.1, let us proceed to adapt it for this test input. As we can see, this test input bears three entity variables Electron22334, NAD-Plus22340 and Electron-Shuttle-System22339 unseen in the training data. We therefore, first create NP rooted trees for each of them, lexicalise with their word forms and associate with their corresponding predicates from the `:INSTANCE-TYPES` section – Figure 3.24a, 3.24b and 3.24c respectively. The variables names have been abbreviated.

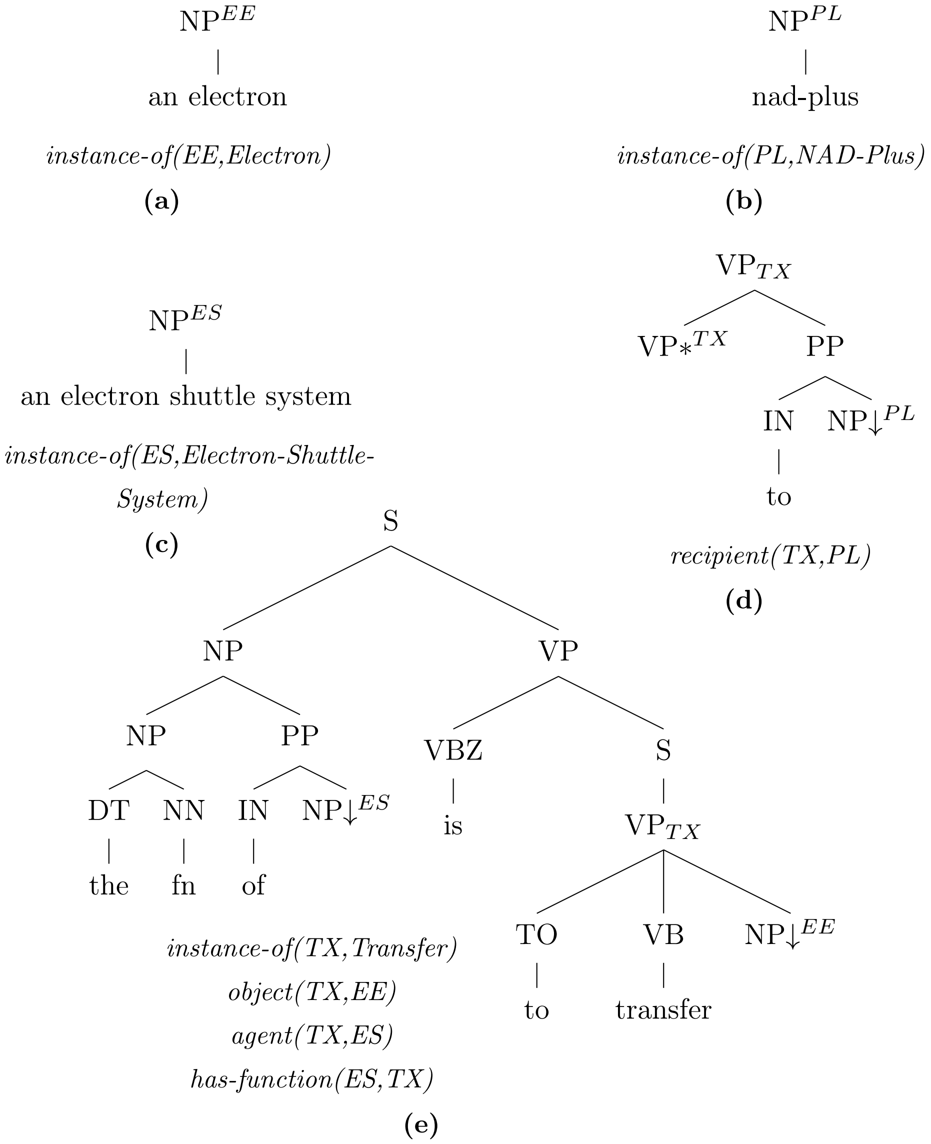


Figure 3.24: Grammar Adaptation for the Test Scenario in Figure 3.23

The event variable Transfer22328 occuring in this test input is not present in our grammar from the training phase. We therefore proceed to build a grammar entry for this test event by adapting the existing grammar. The event variable Transfer22328 is bound to its arguments via the relations object, recipient, agent and has-function. As per our definition of event similarity, we can see that this event in conjunction with its relations object, agent and has-function can be adapted from the tree present in Figure 3.20a (induced from the Training Scenario of Figure 3.1a) and for the relation recipient, it can be adapted from the tree in Figure 3.15e (induced from the different Training Scenario of Figure 3.1b). This gives us two adapted trees in Figure 3.24e and 3.24d respectively; making up the entire semantics of this unseen event variable Transfer22328. Note that the word form verbalising the event variable has been replaced (*transfer* instead of *release* in Figure 3.24e) and the feature structure values have been updated to reflect the variable in the test semantics (both in Figure 3.24d and 3.24e).

Coincidentally, this example also highlights the benefit of the Grammar Expansion activity discussed earlier. Namely, we have benefitted by adapting the trees for the test input from the trees induced from two different training scenarios. In our work, we look for grammar adaptation from the *expanded* grammar only when a correct grammar entry can not be adapted from the *base* grammar; so as to avoid unnecessarily big grammar.

3.5.6 Surface Realisation

To generate with the grammar we have extracted, we use the GenI surface realiser [Gardent *et al.*, 2007]. GenI is an existing surface realiser for Tree Adjoining Grammar. Given an input semantics and a FB-LTAG with a unification based semantics, GenI follows a three-step approach.

First, it selects all the grammar entries whose semantics subsumes the input semantics. In our case, the input semantics consists of all the triples present in the `:TRIPLES` section of the input. When given this input, GenI first selects all the trees from our grammar whose semantics subsumes (part of) the input semantics.

Second, all selected trees are tried for combination using the FB-LTAG combination operations (i.e., adjunction and substitution) as constrained by the unification variables.

Third, the generator outputs the yield of all derived trees which are syntactically complete and whose semantics is the input semantics.

Thus for instance, given the input semantics shown in Figure 3.1a, 3.1c and 3.1b, GenI selects all the trees from Figure 3.14, 3.16 and 3.15 respectively; combines them using FB-LTAG operations and outputs as generated sentence the sentences shown in Figure 3.1a, 3.1c and 3.1b. If more than one derivations generate the same exact sentence (for example, the trees in Figure 3.14 and 3.20 yield the same sentence for the input in Figure 3.1a), GenI avoids the duplicates and yields a single derivation, namely the sentence *The function of a gated channel is to release particles from the endoplasmic reticulum.*

GenI, however, also allows for a `--`*partial* flag, which when present, permits as output the text coming from derived trees covering some portion of the input semantics. That is, in the "partial" mode, we can have ouput sentences that only describe some relations in the input. Note that the derived trees still need to be syntactically complete (i.e. shouldn't have any substitution and adjunction nodes) in order to qualify as successful generation. In our work, we go for partial generation only when the complete generation fails. We defer the discussion of input cases requiring partial generation (e.g. the input in Figure 3.18) to Section 3.7.1. Inputs in Figure 3.1a, 3.1c and 3.1b are all examples of cases that lead to complete generation.

In Figure 3.25 below, we present some examples of sentences generated by our system. We show sample sentences generated from input semantics bearing relations between a single event and its entities (Figure 3.25a), multiple events and entities (Figure 3.25b) and between multiple entities (Figure 3.25c). The word forms verbalising the events are underlined and we put the entities in italics.

Note that several trees in the grammar can subsume some part of semantics for a given input because the trees making our grammar come from different scenarios which may contain overlapping or subset semantics of the given input. Therefore, we can have several different derivations with varying paraphrases as generated sentences for the same input. Thus, we rank and identify the most suitable sentence generated for each input. For this, we train a 3-gram language model on the GeniA corpus[12] using the SRILM toolkit [Stolcke, 2002]. The GeniA corpus is chosen as a training corpus because it contains text from the biology domain which is also the domain of the KB Bio 101 ontology. GeniA consists of 2000 MEDLINE asbtracts about biology containing more than 400000 words [Kim *et al.*, 2003b] and is a widely used corpus for different aspects of NLP from biology texts. Here, we use the language model trained on this corpus to rank our generated sentences by decreasing probability.

[12] `http://www.nactem.ac.uk/genia/`

Relations among a single Event and its Entities

Lysosomal enzymes digest *polymers* in *the lysosome* of *eukaryotic cells*, resulting in *monomers.*
Culture medium as *a nutrient* is a solvent in which *cells* are cultured to produce *culture.*

(a)

Relations among Events and Entities

Radioactive treatment of *cancer* happens with the help of *subatomic particles* emitted from *atomic nucleus* of *radioactive isotope.*
When *a carrier protein* moves through *a biomembrane*, *a hydrophilic compound* detaches from *the binding site.*

(b)

Relations among Entities

The xylem of a *plant vein* has thin *tracheids.*
Algal cells in *algae* have a hollow *rough ER.*

(c)

Figure 3.25: Example generated sentences verbalising different relation types

3.6 Evaluation

Using the induced grammar, we generate sentences for the test input of the KBGen dataset and perform both automatic and human evaluation. As mentioned earlier, the KBGen test set consists of 72 scenarios. Each test scenario, on average, contains 16 triples (including the triples defining relations between variables, their predicates and data types) and 17 words in the corresponding sentence. We have two different benchmarks against which our sentences can be evaluated. This includes the sentences generated by the two other systems (excluding our system described herein) participating in the KBGen challenge. The UDEL system [Butler *et al.*, 2013] is a symbolic handcrafted rule based template system and the IMS system [Zarrieβ and Richardson, 2013] is a statistical system using a probabilistic grammar induced from the training data. The human written sentences for each test scenario provided by the KBGen dataset itself (at the end of the competition) serve as a reference against which our and the sentences from the two benchmarks shall be ranked and we tally the scores for judgement. In Section 3.6.1, we present results of automatic evaluation and in Section 3.6.2, we present the human evaluation.

We evaluate three configurations of our approach on the KBGen test data: one with the *base* grammar we have induced (Base); a second with a manual grammar expansion of the *base* grammar (ManExp); and the third one with the *expanded* grammar obtained after the automated grammar expansion activity (AutExp). Below, we compare the results obtained in these configurations with that present in the benchmarks and report the results.

3.6.1 Automatic Evaluation

For automatic evaluation, we first evaluate coverage. Coverage is the percentage of the total test inputs for which at least one sentence is generated by the respective systems. In the first column of Table 3.26, we mention the coverage of each system and in the second column, we show the number of trees present in various configurations of our grammar. It can be seen that our Base system strongly undergenerates failing to account for 69.5% of the test inputs while both the IMS and the UDEL system have full coverage. However, because our extracted grammar is linguistically principled and relatively compact, it is possible to manually edit it. Indeed, the ManExp results show that, by adding 41 trees to the grammar, coverage can be increased by 52.5 points reaching a coverage of 83%. Finally, the AutExp results demonstrate that the automated expansion mechanism permits achieving full coverage while keeping a relative small grammar (477 trees).

Next, we use the BLEU-4 modified precision score [Papineni *et al.*, 2002] for evaluating the systems; comparing the sentences generated by various systems to the KBGen reference sentences. The average BLEU score is shown with respect to all input (All) and to those inputs for which the systems generate at least one sentence (Covered). In terms of BLEU score, the best version of our system (AutExp) outperforms the probabilistic approach of IMS by a large margin (exceeding by 0.17 units) and produces results similar to the fully handcrafted UDEL system (falling short of 0.03 units).

In sum, our approach permits obtaining BLEU scores and a coverage which are similar to that obtained by a hand crafted system and outperforms a probabilistic approach.

System	Coverage	# Trees	All	Covered
IMS	100%		0.12	0.12
UDEL	100%		0.32	0.32
Base	30.5%	371	0.04	0.39
ManExp	83 %	412	0.28	0.34
AutExp	100%	477	0.29	0.29

Figure 3.26: BLEU scores and Grammar Size (Number of Elementary TAG trees)

3.6.2 Human Evaluation

For the human evaluation, we set up an online survey asking people to rate the sentences generated by three systems – the two benchmarks (UDEL and IMS) and the (AutExp) version of our system since this is the one with the full coverage. The online portal was developed using the LG-Eval toolkit [Kow and Belz, 2012] and participants could use a sliding scale from -50 to +50 to rate the sentences. For each rating, a participant would see a sentence from one of the three systems and the corresponding KBGen reference sentence. The participant is not aware of the system from which the sentence to be rated is taken from and can only compare it with the reference sentence for providing a judgement. We asked the participants to rate each sentence along three dimensions: **Fluency** (Is the text easy to read?), **Grammaticality** (Does the sentence sound natural and linguistically correct?) and meaning similarity or **Adequacy** (Does the meaning conveyed by the sentence correspond to the meaning conveyed by the reference sentence?). A Latin Square Experimental Design was used to ensure that each participant sees the same number of outputs from each system and for each test set item. By moving the slider from -50 (meaning total disagreement) to +50 (meaning complete acceptance) to answer each question, the participants provided their overall ratings (scores). 12 subjects participated in the evaluation and each sentence was rated by 3 participants.

In Figure 3.27, we report the results of human evaluation. Figure 3.27a shows the analysis of the systems for their **Fluency**, Figure 3.27b for **Grammaticality** and Figure 3.27c for the **Adequacy** measures. For all these criteria, we present the mean score (Mean) obtained for sentences of each system (on a scale of 0 to 5), the grading of systems (Homogeneous Subsets) organized by letters (A, B or C) denoting their ANOVAs (Analysis of Variance) with post-hoc Tukey significance of p<0.05 and the measure of standard deviation (Standard Deviation) as observed in the scores obtained for each system. The grading signifies that the systems rated with the same grade are homogeneous in the confidence interval of p<0.05 and the standard deviation is an estimate of the variation observed in the scores provided by the participants.

Fluency			
System	Mean	Homogeneous Subsets	Standard Deviation
UDEL	4.36	A	0.95
AutExp	3.45	B	1.5
IMS	1.91	C	1.56

(a)

Grammaticality			
System	Mean	Homogeneous Subsets	Standard Deviation
UDEL	4.48	A	0.8
AutExp	3.55	B	1.36
IMS	2.05	C	1.65

(b)

Meaning Similarity			
System	Mean	Homogeneous Subsets	Standard Deviation
UDEL	3.69	A	1.4
AutExp	3.65	A	1.48
IMS	1.31	B	1.3

(c)

F-ratios		
Fluency	Grammaticality	Meaning Similarity
68.58	71.94	79.55

(d)

Figure 3.27: Human Evaluation Results on a scale of 0 to 5. Homogeneous subsets are determined using Tukey's Post Hoc Test with p<0.05.

The homogeneous subsets analysis reveals significant differences among the competing systems. In terms of **Fluency** and **Grammaticality**, our system is closer to the handcrafted UDEL system (margin of 0.91 and 0.93 units respectively) than it departs from the statistical IMS system (margin of 1.54 and 1.50 units respectively). In terms of **Adequacy**, it ranks on the same level as the UDEL system (belonging to the same homogeneous set A with the margin of 0.04 units) and distances itself from the IMS system with a very wide margin (2.34 units). Thus, our system consistently ranks second – behind the handcrafted UDEL system and before the statistical IMS; confirming the ranking based on BLEU.

The standard deviation values by themselves do not represent the quality of the systems evaluated but are indicative of the distribution of scores observed for each system and are used for the computation of F-ratios (shown in Figure 3.27d). The relatively high values of F-ratios indicate that the mean values obtained (and subsequently the homogeneous subsets) are the effect of consistent score patterns and not likely due to chance.

3.7 Discussion

A key feature of the grammar we have induced is that it respects the linguistic principles of Tree Adjoining Grammar namely,

Lexicalisation: Each elementary tree should has at least one lexical item. In our trees, we always have an anchor node and possibly many co-anchors.

Extended Domain of Locality: Each elementary tree contains all (and only) the arguments slots of the syntactic functor it represents.

Elementary Tree Minimality [Frank, 2002]: Each elementary tree should capture a single semantic unit.

However, some practical issues arising from this work call for additional research. These issues have only been partially addressed in this work and in sections below, we discuss them in turn. We highlight the current remedies and provide pointers for further research direction.

3.7.1 Partial AND/OR Erroneous Generation

For some inputs, multiple relations share the same entity variables. Let us consider the example in Figure 3.18. Here, we have two relations, namely the `(|Plant-Vein3268| |has-part| |Xylem3269|)` and `(|Xylem3269| |has-part| |Tracheid3271|)` that share the same entity variable `Xylem3269`. As already discussed, our grammar extraction approach is modeled after capturing relations and these relations are of four different types – event-to-entity, event-to-event, entity-to-event and entity-to-entity (as seen in Section 3.3). Fundamentally, the entities are the building blocks making up the relations (the event-to-event relations associate events which in turn are composed of event-to-entity relations) and when they are shared across the relations, it leads to extraction of trees which cannot be fully combined (semantically plus syntactically) to obtain the correct reference sentence. In other words, we shall either have an erroneous generation arising out of the fully combined trees or only some trees can be combined (i.e. partial generation) which may or may not generate a complete sentence.

The trees extracted for the input in Figure 3.18 depict such an instance. For generation from this input, GenI selects all the trees extracted – those in Figure 3.18b, 3.18c, 3.18d, 3.18e, 3.18f and 3.18g making up for the complete semantics in the input. With all these trees, a syntactically complete generation can occur as follows. The initial trees in Figure 3.18d and 3.18e can be substituted to the left and right substitution nodes respectively of the tree in Figure 3.18b and a derived tree, say d_1, is obtained. Another derived tree, say d_2, can be obtained by adjoining the tree in Figure 3.18g to the one in Figure 3.18f. Then the trees d_1 and d_2 can be substituted at the left and right substitution nodes of the tree in Figure 3.18c and a complete (both semantically and syntactically) generation is realised, outputting the sentence "*The xylem of a plant vein of a plant vein has thin tracheids.*", which is grammatically incorrect sentence.

On the other hand, a grammatically correct sentence could be obtained for the same input in a partial generation setting. The tree in Figure 3.18d and the derived tree d_2 (obtained as earlier) can be substituted for syntactic completeness in the left and right substitution nodes of the tree in Figure 3.18c leading to a grammatically correct sentence "*The xylem of a plant vein has thin tracheids.*". This however, leaves out the semantic information associated to trees in Figure 3.18b and 3.18e; thus an instance of partial generation.

For the example input just discussed, we have shown cases of complete generation with ungrammatical ouput and partial generation with grammatically correct output. This is however not always the case for relations sharing entity variables. In particular, the complete generation itself may not be possible because of syntactic combination incompatibility of extracted trees (thus no complete generation at all) or the partial generation may output incomplete sentences. In any case, the sentences so generated are not true reflection of their input and therefore additional measures should be considered to address such inputs.

3.7.2 Multiple Adjunctions Order

For generation resulting from a combination of many adjoining trees, the grammar we have extracted does not define the order of such combination. That is, although our grammar constrains the trees that can adjoin to a given node (through the feature values composed of variables occurring in the semantics), it allows for their adjunction in any possible order. Let us consider the trees in Figure 3.21, obtained after the grammar expansion phase for the input in Figure 3.1b. As we can see, the subtrees in Figure 3.21d, 3.21b and 3.21c can adjoin to the NP node bearing the same feature value DA in the subtree of Figure 3.21a in any order. Thus, from this grammar (and including the entity subtrees from its *base* grammar in Figure 3.15) we can have as output the following sentences :

(8) *Concentration gradient provides energy for the diffusion of anions from a cell to the extracellular matrix.*

(9) *Concentration gradient provides energy for the diffusion to the extracellular matrix from a cell of anions.*

and many more resulting from the different combination order of the adjoining trees. As we can see, such combinations contribute to overgeneration and also semantically anomalous sentences (e.g. Sentence 9).

A possible strategy to avoid this problem would be to additionally constrain the trees in our grammar with more feature values. One could, for instance, add feature values depicting the verbalisation order of combining relations as seen in the training scenarios. This would help to improve the grammatical correctness, fluency as well as adequacy of the generated sentences.

3.7.3 Ranking

As discussed earlier, we can have several different derivations with varying paraphrases as generated sentences for the same input. In this work, we ranked the generated sentence using a 3-gram language model. Another possibility, and perhaps better, would be rank the individual trees selected for the given input before they are combined via TAG operations. For example, a beam search using the same language model could be used for identifying the k-best combination of trees, instead. Following this, the surface realiser would only take those tree combinations as input and give us k-best sentences as ouput. This strategy also reduces the computational cost (both time and space) of generation since the number of trees to be used for generation is minimized.

3.8 Conclusion

We have presented a supervised approach to grammar based generation from Knowledge Bases. Using the KBGen dataset, we proposed a novel method for corpus-based learning of grammar that can mediate the semantic-syntactic mapping of meaning representation in KB triples to strings in sentences. We have shown that the resulting grammar is linguistically principled and compares favorably with competing symbolic and statistical approaches. At the same time, it is also generic and can be adapted for generation from any knowledge base data.

Chapter 4

Verbalising n-Ary Events in Ontologies – A Weakly Supervised Approach

Contents

This chapter is based on the following publications :

Bikash Gyawali, Claire Gardent and Christophe Cerisara. *Automatic Verbalisation of Biological Events*, in Third International Workshop on Definitions in Ontologies (IWOOD), Lisbon, Portugal, July 2015

Bikash Gyawali, Claire Gardent and Christophe Cerisara. *A Domain Agnostic Approach to Verbalizing n-ary Events without Parallel Corpora* in Proceedings of the 15th European Workshop on Natural Language Generation (ENLG), pages 18 – 27, Brighton, UK, September 2015

In this chapter, we present a weakly supervised approach for surface realisation from Knowledge Bases. Taking the KBGen dataset as reference, we present a novel approach to surface realisation in terms of generating descriptions of events in n-ary relation to arguments in ontologies. We propose a generic, domain-independent, probabilistic method which extracts event verbalisation frames from large domain corpora and uses probabilities both to select an appropriate frame and to map between syntactic and semantic arguments, i.e. to determine which event argument fills which syntactic function (e.g., subject, object) in the selected frame. We evaluate our approach on a corpus of 336 event descriptions, provide a qualitative and quantitative analysis of the results obtained and discuss possible directions for further work.

4.1 Introduction

While earlier work on data-to-text generation heavily relied on handcrafted linguistic resources, more recent data-driven approaches have focused on learning a generation system from parallel corpora of data and text. Thus [Angeli *et al.*, Chen and Mooney, Wong and Mooney, Konstas and Lapata, Konstas and Lapata, 2010, 2008, 2007, 2012b, 2012a] trained and developed data-to-text generators on datasets from various domains including the air travel domain [Dahl *et al.*, 1994], weather forecasts [Liang *et al.*, Belz, 2009, 2008] and sportscasting [Chen and Mooney, 2008]. In both cases, considerable time and expertise must be spent on developing the required linguistic resources. In the handcrafted, symbolic approach, appropriate grammars and lexicons must be specified while in the supervised approach, an aligned data-text corpus must be built for each new domain. To overcome this shortcoming, we propose an alternative, a weakly supervised approach to surface realisation from knowledge bases which could be used for any knowledge base for which there exists large textual corpora.

A more specific, linguistic, issue which has received relatively little attention is the non-supervised verbalisation of n-ary relations and the task of appropriately mapping KB roles to syntactic functions.

In recent work on verbalising RDF triples, relations are restricted to binary relations (called “property” in the RDF language) and the issue is therefore intrinsically simpler.

In symbolic approaches dealing with n-ary relations, the mapping between syntactic and semantic arguments is determined by the lexicon and must be manually specified.

Finally, in data-driven approaches, the mapping is learned from the alignment between text and data and is restricted by cases seen in the training data.

Instead, we learn a probabilistic model designed to select the most probable mapping. In this way, we provide a domain independent, fully automatic, means of verbalising n-ary relations.

This chapter is organized as follows. Section 4.2 provides a survey of related work. In Section 4.3, we introduce the KBGen$^+$ dataset which we use for our experiments in this work. Section 4.4 describes our approach, discussing the various subtasks (Corpus Collection in 4.4.1, Lexicon Creation in 4.4.2, Frame Extraction in 4.4.3, Probabilistic Models in 4.4.4 and Surface Realisation in 4.4.5) in detail. Next, we present the results and their evaluation in Section 4.5 and in Section 4.6, we discuss the problem cases, limitations and possible remedies. Section 4.7 concludes.

4.2 Related Work

As already mentioned in Section 3.2, there has been much research in recent years on developing natural language generation systems which support verbalisation from knowledge and data bases. We now review how these approaches address the points that were just mentioned, namely, the need to minimize the amount of manual work required to build the necessary linguistic resources; and the verbalisation of n-ary relations together with the appropriate linking between syntactic and semantic arguments.

Many of the existing KB Verbalising tools rely on generating so-called Controlled Natural Languages (CNL) i.e., a language engineered to be read and written almost like a natural language but whose syntax and lexicon is restricted to prevent ambiguity. For instance, the OWL verbaliser integrated in the Protégé tool is a CNL based generation tool, [Kaljurand and Fuchs, 2007] which provides a verbalisation of every axiom present in the ontology under consideration. Similarly, [Wilcock, 2003] describes an ontology verbaliser using XML-based generation. Finally, recent work by the SWAT project[13] has focused on producing descriptions of ontologies that are both coherent and efficient [Williams and Power, 2010]. In these approaches, the mapping between relations and verbs is determined either manually or through string matching. Moreover, KB relations are generally assumed to map to binary verbs.

More complex NLG systems have also been developed to generate text (rather than simple sentences) from knowledge bases. Thus, the MIAKT project [Bontcheva

[13] http://crc.open.ac.uk/Projects/SWAT

and Wilks., 2004] and the ONTOGENERATION project [Aguado *et al.*, 1998] use symbolic NLG techniques to produce textual descriptions from some semantic information contained in a knowledge base. Both systems require some manual input (lexicons and domain schemas). More sophisticated NLG systems such as TAILOR [Paris, 1988], MIGRAINE [Carenini *et al.*, 1994], and STOP [Reiter *et al.*, 2003] offer tailored output based on user/patient models. While offering more flexibility and expressiveness, these systems are difficult to adapt by non-NLG experts because they require the user to understand the architecture of the NLG systems [Bontcheva and Wilks., 2004]. Similarly, the NaturalOWL system [Galanis *et al.*, 2009] has been proposed to generate fluent descriptions of museum exhibits from an OWL ontology. These approaches however rely on extensive manual annotation of the input data.

Related to the work discussed in this paper is the task of learning subcategorization information from textual corpora. Automatic methods for subcategorization frame acquisition have been proposed from general text corpora, e.g., [Briscoe and Carroll, 1997], [Korhonen, 2002], [Sarkar and Zeman, 2000] and specific biomedical domain corpora as well [Rimell *et al.*, 2013]. Such works are limited to the extraction of syntactic frames representing subcategorization information. Instead, we focus on relating the syntactic and semantic frame and, in particular, on the linking between syntactic and semantic arguments.

We take the insipiration for linking of syntactic and semantic arguments from the probabilistic mapping approaches proposed in works related to semantic role labeling, such as in [Swier and Stevenson, 2004]. There, the authors present an approach for inducing semantic roles for verb arguments using an existing verb lexicon, the VerbNet [Kipper *et al.*, 2000]. VerbNet is a computational verb lexicon that specifies the set of syntactic frames for verbs along with the semantic roles that label the syntactic slots occuring in those frames. Given a verb occuring in a sentence, [Swier and Stevenson, 2004] select the entries from the VerbNet lexicon which provide a near match to the syntactic structure of this verb. Then, using a backoff probability model which relies on the frequency of observations seen in the lexicon for the specific combination of the given verb and its syntactic slots, they induce the most probable semantic role to assign for each syntactic arguments of the given verb. In our case, we extract the syntactic frames as observed in the sentences of the corpus and the semantic roles binding the arguments are present in the input. We identify the syntactic frames which verbalise the semantic roles of the input and build probabilistic models of syntax/semantics linking taking into account the frequency of such observations.

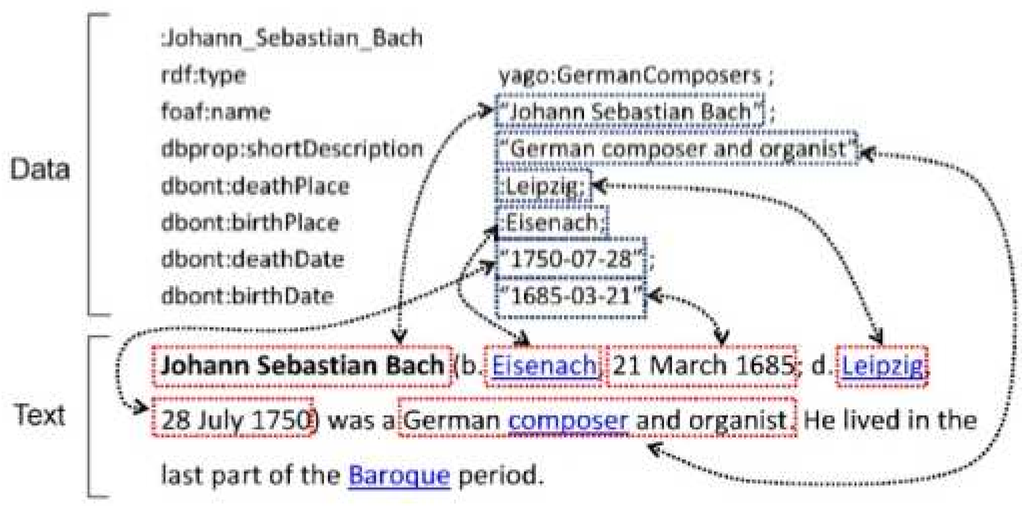
Another trend of work relevant to this paper is generation from databases using parallel corpora of data and text. [Angeli *et al.*, 2010] train a sequence of discriminative models to predict data selection, ordering and realisation. [Wong and Mooney, 2007] uses techniques from statistical machine translation to model the generation task and [Konstas and Lapata, 2012b, Konstas and Lapata, 2012a] learns a probabilistic Context-Free Grammar modelling the structure of the database and of the associated text. Various systems from the KBGen [Banik *et al.*, 2012, Banik *et al.*, 2013] shared task – [Butler *et al.*, 2013], [Gyawali and Gardent, 2013] and [Zarrieβ and Richardson, 2013] perform generation from the same input data source as ours and use parallel text for supervision. Our approach differs from all these approaches in that it does not require parallel text/data corpora.

Finally, there has been much work recently on extracting templates from comparable corpora. Such approaches are based on finding sentences in the corpus that bear verbalisation of all the data in the given input. Once such a sentence is found for an input, a template is constructed from it by replacing the word forms verbalising the data with slots marked by the variables making up those data in the input. Figure 4.1 adapted from [Duma and Klein, 2013] depicts this process. The arrows show the alignment of data in the input to strings in a sentence of the text. As we can see the sentence verbalises all the data occuring in the input (except for the value of the rdf:type variable which is, in fact, just a type identifier and doesn't represent a verbalizable data unit). A template as shown in Figure 4.1b is then extracted by replacing the aligned strings with the variables in the input holding the respective data. In the template, we show the variables making up the slots in bold.

The works described in [Kondadadi *et al.*, 2013], [Ell and Harth, 2014] and [Duma and Klein, 2013] are all examples of template extraction approach and require comparable text corpora that are closely related to the knowledge base which is being generated from. In practice however, we found that despite the large size of the training corpus we built for KBGen, it was not possible to extract templates from this corpus which would succeed in directly matching all the data in a given input to surface text in the sentences obtained from non-parallel biomedical texts. Instead, we therefore guess (using probabilities) a lemma and a subcategorization frame for verbalising the event and we learn the linking between semantic and syntactic arguments.

4.3 The KBGen$^+$ Dataset

The dataset for this experiment is derived from the KBGen dataset discussed earlier in Chapter 3.



Johann Sebastian Bach (b. Eisenach, 21 March 1685; d. Leipzig, 28 July 1750) was a German composer and organist. He lived in the last part of the Baroque period.

(a) Template Extraction from Comparable Corpora as shown in [Duma and Klein, 2013]

Name (b.**birthPlace**, **birthDate**; d. **deathPlace**, **deathDate**) was a **shortDescription**.

(b) Template Created for the above input

Figure 4.1: An example of Corpus based Template Extraction

In KBGen the input consists of content units, each of which expresses a set of relations among different concept types, namely event-to-entity, event-to-event, entity-to-event, entity-to-entity and property-values relations. In this work, however, we are interested in describing the events in relation to their entity type arguments only and, therefore, we process the KBGen dataset to produce all KB fragments which represent a single event with roles to entities only. In other words, given the content units from the KBGen dataset, we automatically filter out event-to-event, entity-to-event, entity-to-entity and property-values relations and keep only the event-to-entity relations so as to obtain the dataset for this experiment (dubbed KBGen$^+$ henceforth). The KBGen$^+$ dataset is thus a collection of biological event descriptions whereby an *event description* consists of an event, its arguments and the roles relating each argument to the event. The events are concepts of type Event (e.g., Release), arguments are concepts of type Entity (e.g., Gated-Channel, Vascular-Tissue, Iron) and roles are relations from events to entities (e.g., agent, patient, path, instrument). Note that more than one event descriptions can sometimes be produced from a single content unit of the KBGen dataset. For example, given the following content unit of KBGen :

```
:TRIPLES (
(|Detach10084| |object| |Hydrophilic-Compound10085|)
(|Detach10084| |site| |Functional-Region10079|)
(|Move-Through10088| |subevent| |Detach10084|)
(|Move-Through10088| |base| |Biomembrane10093|)
(|Move-Through10088| |agent| |Carrier-Protein10080|))

:INSTANCE-TYPES (
(|Hydrophilic-Compound10085| |instance-of| |Hydrophilic-Compound|)
(|Functional-Region10079| |instance-of| |Binding-Site|)
(|Detach10084| |instance-of| |Detach|)
(|Biomembrane10093| |instance-of| |Biomembrane|)
(|Move-Through10088| |instance-of| |Move-Through|)
(|Carrier-Protein10080| |instance-of| |Carrier-Protein|))

:ROOT-TYPES (
(|Detach10084| |instance-of| |Event|)
(|Hydrophilic-Compound10085| |instance-of| |Entity|)
(|Functional-Region10079| |instance-of| |Entity|)
(|Move-Through10088| |instance-of| |Event|)
(|Biomembrane10093| |instance-of| |Entity|)
(|Carrier-Protein10080| |instance-of| |Entity|))
```

we obtain the following two isolated event descriptions shown below. For ease of reading, we shall use the `:INSTANCE-TYPES` values of the variables in the event descriptions (such as `Detach` for the event variable `Detach10084`, `Biomembrane` for the entity variable `Biomembrane10093` in the event descriptions below) in the remainder of this text. Further, we shall present only the triples in the `:TRIPLES` section of the input and organize them such that each triple statement is in the fashion `(Event role Entity)`.

Event Description 1

```
(|Detach| |object| |Hydrophilic-Compound|)
(|Detach| |site| |Functional-Region|)
```

Event Description 2

```
(|Move-Through| |base| |Biomembrane|)
(|Move-Through| |agent| |Carrier-Protein|)
```

In total, we obtain 336 event descriptions for our KBGen$^+$ dataset. Various statistics for this dataset are shown in Table 4.1 below. The counts represent the total number of occurrences of the corresponding item in the KBGen$^+$ dataset while the numbers in parenthesis correspond to their distinct counts. As we can see from the table, on average, event descriptions have more than 2 roles. Further analysis shows that 4.46% (count 15) of the total event descriptions have exactly 1 role (the minimum), 0.5% (count 2) have exactly 8 roles (the maximum), 47.61% (count 160) have exactly 2 roles, 23.51% (count 79) have exactly 3 roles and 24.40% (count 82) of the total event descriptions have more than 3 roles. Also, from the Table 4.1, we can see that in total we have 397 (= 126 + 271) distinct variables in the KBGen$^+$ dataset.

Items	**Count**
Total nb of Event Descriptions	336
Min/Avg/Max nb of roles in an Event Description	1/2.76/8
Total nb of events	336 (126)
Total nb of entities	929 (271)
Total nb of roles	929 (14)

Table 4.1: KBGen$^+$ Statistics.

4.4 Methodology

Our goal is to automatically generate natural language verbalisations of the event descriptions in the KBGen$^+$ dataset. For this, we propose a probabilistic method which extracts possible verbalisation frames from large biology specific domain corpora and uses probabilities both to select an appropriate frame given an event description and to determine the mapping between syntactic and semantic arguments. That is, probabilities are used to determine which event argument fills which syntactic function (e.g., subject, object) in the produced verbalisation. In sections below, we describe each of these steps in greater detail.

4.4.1 Corpus Collection

We begin by gathering sentences from several of the publicly available biomedical domain corpora.[14] This includes the BioCause [Mihăilă *et al.*, 2013], BioDef[15], BioInfer [Pyysalo *et al.*, 2007], Grec [Thompson *et al.*, 2009], Genia [Kim *et al.*, 2003a] and PubMedCentral (PMC)[16] corpus. We also include the sentences available in annotations of named concepts in the KB Bio 101 ontology. This custom collection of sentences will be the corpus upon which our learning approach will build on. Table 4.2 lists the count of sentences available in each corpus and in total.

	Number of Sentences
BioCause	3,187
BioDef	8,426
BioInfer	1,100
Genia	37,092,000
Grec	2,035
PMC	7,018,743
KBBio101	3,393
Total	44,128,884

Table 4.2: Count of sentences in different corpora

4.4.2 Lexicon Creation

To identify corpus sentences which might contain verbalisations of KBGen$^+$ events and entities, we build a lexicon mapping the event and entity variables contained in KBGen$^+$ to natural language words or phrases using several existing resources.

4.4.2.1 KBGen

First, we take the lexicon provided by the KBGen challenge. As discussed in Chapter 3, the KBGen lexicon is composed of entries that provide inflected forms and nominalizations for the event variables (for example, Block) and singular and plural noun forms for the entity variables (for example, Earthworm), such as :

[14] Ideally, since KB Bio 101 was developed based on a textbook, we would use this textbook as a corpus. Unfortunately, the textbook, previously licensed from Pearson, is no longer available.

[15] Obtained by parsing the ⟨Supplement⟩ section of html pages crawled from `http://www.biology-online.org/dictionary/`

[16] `ftp://ftp.ncbi.nlm.nih.gov/pub/pmc`

KB Symbol	**Word forms**
Block	*blocks, block, blocked, blocking*
Earthworm	*earthworm, earthworms*

At this phase, we simply copy the entries present for the KBGen$^+$ variables in the KBGen lexicon and create our lexicon.

4.4.2.2 Mesh and BioDef

To the lexicon so obtained, we add the synonymous entries for KBGen$^+$ events and entities found in the Mesh[17] and in the BioDef vocabulary. Mesh is an existing wide-coverage thesaurus of terms in life sciences and provides term synonymy. BioDef is our custom name for the synonyms vocabulary we build automatically by parsing the entries in ⟨Synonyms⟩ section of html pages crawled from an open biology dictionary at `http://www.biology-online.org/dictionary/`. Some example synsets obtained from Mesh and BioDef are shown below:

block, prevent, stop
neoplasm,tumors,neoplasia,cancer

We use these synsets to augment our lexicon as follows. For each event and entity variables, we look for synsets that contain one or more word forms already assigned to that variable in our existing lexicon. If such a synset is found for a variable, we add all the terms present in that synset as lexical entries for that variable, avoiding duplicates. Thus, for the example above, our lexicon would be updated as shown below :

KB Symbol	**Word forms**
Block	*blocks, block, blocked, blocking, prevent, stop*
Earthworm	*earthworm, earthworms*

4.4.2.3 KBBio101

Finally, for generalisation purposes, we automatically extract the direct parent and siblings of the KBGen$^+$ events and entities in the KB Bio 101 ontology and add them as a lexical entries for the corresponding KBGen$^+$ event/entity. For example, for the KB Bio 101 event Block, the direct parent and siblings extracted from the KB Bio

[17] `http://www.nlm.nih.gov/mesh/filelist.html`

101 are, respectively:

make–inaccessible
conceal, deactivate, obstruct

and we update the lexical entries for the event Block with these word forms[18] in order to get an updated lexicon as follows :

KB Symbol	**Word forms**
Block	*blocks, block, blocked, blocking, prevent, stop, conceal, deactivate, obstruct*
Earthworm	*earthworm, earthworms*

The resulting lexicon is thus a merge of all the entries extracted from all the above mentioned sources for all the KBGen$^+$ events and entities. In Table 4.3 below, we present the size of lexicon available from each source (Total Entries) and the count of distinct KBGen$^+$ variables (events plus entities) for which one or more entry was found in that source (Intersecting Entries). Table 4.4 details this by showing the proportion of KBGen$^+$ events and entities for which a lexical entry was found in each source. At a first glace, it might seem redundant to use both the **KBGen** and **KBBio101** sources since we obtain a 100% lexical coverage for both the event and entity variables from each of them. However, in practice, they contribute to different word forms (e.g. *block* is obtained from **KBGen** and *obstruct* is obtained from **KBBio101** for the event Block) and this is crucial for our task (as discussed in the following sections) which is based on using syntax/semantic associations as observed in varying syntactic environments in which the different words occur in the sentences of the corpus.

	Total Entries	Intersecting Entries
KBGen	469	397
Mesh	26795	65
BioDef	14934	99
KBBio101	6972	397

Table 4.3: Total number of lexical entries and the number of distinct KBGen$^+$ variables (events plus entities) observed in each source

In Table 4.5, we present the information on the maximum, minimum and average number of lexical items available for KBGen$^+$ event and entities in the merged lexicon.

[18] To avoid overly general classes, we exclude hyphenated and multi-word terms.

	KBGen	Mesh	BioDef	KBBio101	ALL
Event	100%	10.31%	25.39%	100%	100%
Entity	100%	19.18%	24.72%	100%	100%
All	100%	16.37%	24.93%	100%	100%

Table 4.4: Proportion of KBGen$^+$ Events and Entities for which a lexical entry was found

As we can see, by using multiples sources for lexicon creation, we have on average 18 different word forms verbalising the variables in our input. Having such a big lexicon significantly contributes to extracting a large collection of different syntactic frames for the events (Section 4.4.3) and suggests plausible syntax/semantics mapping for verbalisation of event-entity relations (Section 4.4.4).

	Minimum	Maximum	Average
Event	5	97	22
Entity	3	91	16.18
All	3	97	18.03

Table 4.5: Minimum, maximum and average number of lexical items available for KBGen$^+$ events and entities in the merged lexicon

4.4.3 Frame Extraction

Events in KBGen$^+$ take an arbitrary number of participants ranging from 1 to 8. Knowing the lexicalisation of an event is therefore not sufficient. For each event lexicalisation, information about syntactic subcategorisation and syntactic/semantic linking is also required. Consider for instance, the following event representation:

```
(|Block| |instrument| |PC/EBP|)
(|Block| |object| |TNF-activation|)
(|Block| |base| |Myeloid-Cells|)
```

Knowing that a possible lexicalisation of the event Block is the finite verb form *blocked* (as given by the lexicon) is not sufficient to produce an appropriate verbalisation of the KB event e.g.,

(10) *C/EBP beta blocked TNF activation in myeloid cells.*

In addition, one must know that this verb (i) takes a subject, an object and an optional prepositional argument introduced by a locative preposition (subcategorisation information) and (ii) that the instrument role is realised by the subject slot, the

OBJECT role by the object slot and the BASE role by the prepositional slot (syntax/semantics linking information). That is, we need to know, for each KB event e and its associated roles (i.e., event-to-entity relations), first, what are the syntactic arguments of each possible lexicalisations of e and second, for each possible lexicalisation, which role maps to which syntactic function.

To address this issue, we extract syntactic frames from our constructed corpus and we use the collected data to learn the mapping between the KB and the syntactic arguments.

Frame extraction proceeds as follows. For each event e in the KBGEN$^+$ dataset, we look for all the sentences S in the corpus that mention one or more of the word forms available for this event in the merged lexicon. Each of those sentences $s \in S$ is then parsed using the Stanford dependency parser[19] for collapsed dependency structure. From the resulting dependency parse tree, we extract the subtree t rooted at the node labelled with the word form for the event variable and spanning just its immediate dependents (i.e. the direct children nodes). The frame obtained for the event e from this sentence s is then a string composed of ordered sequence of dependency relations occuring in t along with the part-of-speech (pos) tag of the root node. In the frame, we generalise the pos tags NN, NNS, NNP and NNPS as NP; the pos tags VBD, VBG, VBN, VBP and VBZ as VB and preserve the rest as such. We shall call a frame having the post tag X as a X rooted frame. Thus, given the sentence and the dependency tree shown in Figure 4.2, the extracted frame for the event BLOCK will be a VB rooted frame as follows :

nsubj,VB,dobj

indicating that the verb form *block* requires a subject and an object.

That is, a frame describes the arguments required by the lexicalisations of an event and the syntactic function they realise (e.g. the verb subcategorisation information).

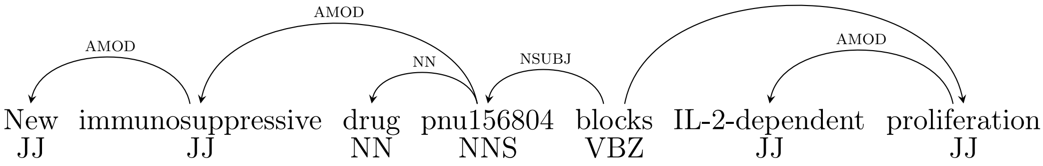


Figure 4.2: Example Dependency Parse Tree

[19] http://nlp.stanford.edu/software/lex-parser.shtml

When extracting the subtrees making up the frames, we only consider a subset of the dependency relations produced by the Stanford parser to avoid including in the frame adjuncts such as temporal or spatial phrases which are optional rather than required arguments. Specifically, the dependency relations we consider for frame construction are:

agent, amod, dobj, nsubj, nsubjpass, prep_ across, prep_ along, prep_ at, prep_ inside, prep_ down, prep_for, prep_from, prep_ in, prep_ away_from, prep_ into, prep_ with, prep_ out_ of, prep_ through, prep_ to, prep_ toward, prep_ towards, prep_ via, prep_ of, auxpass, vmod_ creating, vmod_forming, vmod_producing, vmod_ resulting, vmod_ using, xcomp_ using.[20]

A total of 718 distinct event frames were observed whereby 97.63% of the KBGen^{+} events were assigned at least one frame and each event was assigned an average of 82.01 distinct frames. Several points are worth noting:

- Many different frames can be obtained for the same event. This can arise from the same or different word forms (corresponding to the event variable in the lexicon) occurring in different syntactic environments. For example, both Sentence 11 and 12 verbalise the event Block using the same word form *block* but give rise to two distinct frames, “nsubj,VB,dobj” and “nsubj,VB,dobj,prep_at” respectively. Additionally, from Sentence 13, we can extract a different frame “VB,dobj,prep_in”; this time using a different word form *prevent* present in the lexicon for the same event Block.

  (11) *Studies revealed that grks block excess stimulus.*

  (12) *Alternately, paralytic drugs block synaptic transmission at neuromuscular junctions.*

  (13) *Intervention studies are needed to prevent functional decline in this high-risk population.*

- The same frame can be observed multiple times for a given event. For example, given two distinct sentences, Sentence 14 and 15 below, we observe the same frame “nsubj,VB,dobj” for the event *Absorb* in both of them.

[20] *vmod_ creating, vmod_forming, vmod_ producing, vmod_ using, xcomp_ using, vmod_ resulting* are not directly given by the Stanford parser but reconstructed from a *vmod* or an *xcomp* dependency “collapsed” with the lemmas *producing* or *using* much in the same way as the *prep_ P* collapsed dependency relation provided by the Stanford Parser. These added dependencies are often used in biomedical text to express e.g., RESULT or RAW-MATERIAL roles.

(14) *The chlorophyll best absorbs light in the blue portion of the electromagnetic spectrum.*

(15) *Molecules absorb photon.*

- Different events can subcategorise for the same frame. For example, in Sentence 16 and 17 below, two different events *Secrete* and *Store* have the same frame value “nsubj,VB,dobj”.

(16) *Pancreas cells secrete digestive enzyme.*

(17) *Adipose cells store fat molecules.*

Given an event E, and a sentence S, Algorithm 1 outlines the complete steps involved in our approach for extracting all the frames $\mathcal{F}$ that can be deduced for that event from that sentence. For each event in the KBGen$^+$ dataset, we run this algorithm over all the training sentences; collect the frames so obtained and assign them to the respective event.

4.4.4 Probabilistic Models

The frame extraction phase helps to learn the syntactic verbalisation patterns for the events in the KBGen$^+$ dataset. However, the variety of frames learnt for each event can be very high due to the wide diversity of lexical and syntactic verbalisations present in the training corpus. Also, not all frames learnt for an event occur with equal frequencies or in similar contexts in the training corpus. It is therefore necessary to establish some judgement criteria for identifying the frames most suitable to verbalise a given event. Furthermore, we need to establish the syntactic/semantic mapping between the syntactic structure in frames to the semantic roles in the KBGen$^+$ dataset for a successful realisation system.

To address these issues, we propose three different probabilistic models that are trained on the extracted frames and will be used for generating KBGen$^+$ events descriptions during test time. Given F a set of syntactic frames, E a set of KBGen$^+$ events, D a set of syntactic dependency names and R, a set of KB roles, we discuss each of the three probability models, namely the $P(f|e)$, $P(f|r)$ and $P(d|r)$ in the following sections. All these models we propose are generative probability models and we use a Symmetric Dirichlet prior with hyperparameter $\alpha = 0.1$ in order to counterbalance sparse distributions.

Algorithm 1 Frame Extraction Algorithm

```
function COMPUTEFRAME(Event $E$, Sentence $S$)
    $\mathcal{F} = [\ ]$                                              // Initialised to empty array
    $X = \{$agent, amod, dobj, nsubj, nsubjpass, prep_across, prep_along, prep_at,
      prep_away_from, prep_down, prep_for, prep_from, prep_in, prep_inside,
      prep_into, prep_of, prep_out_of, prep_through, prep_to, prep_toward,
      prep_towards, prep_via, prep_with, vmod_creating, vmod_forming,
      vmod_producing, vmod_resulting, vmod_using, xcomp_using, auxpass$\}$
    $P$ = Dep_Parse($S$)                    // Get Dependency Parse of $S$ using Stanford Parser
    $Q$ = Heads($P$)                                  // Get All head words in the parse tree $P$
    for each $W$ in CONTAINED_IN(E,S) do
        for each $H$ in $Q$ do
            if $H == W$ then
                $D$ = Dependents($H$)            // Get All dependents of H using Stanford Parser
                Add $H$ to $D$
                sort($D$)                                                // Sort by node number
                $f$ = “”                                            // Initialise to empty string
                for each dependent $N$ in $D$  do
                    if $N == H$ then
                                        $f = f$ + GET_GPT(N) + “,”
                    else
                        $dep$ = Dependency($H, N$)        // Dependency relation from $H$ to $N$
                        if $dep \in X$ then
                                        $f = f + dep$ +“,”
                        end if
                    end if
                end for
                $\mathcal{F} \leftarrow \mathcal{F} \uplus \{\{f\}\}$
            end if
        end for
    end for
    return $\mathcal{F}$
end function

function CONTAINED_IN($E, S$)
    $\mathcal{V} = \{\}$                                                          // Empty Set
    $W$ = Words($S$)                                      // Get All words in the sentence
    $lex$ = Lexicon{:E}                          // Get All lexical items of E from Lexicon
    for each $L$ in $lex$ do
        $T$ = Words($L$)                                               // Get All words in $L$
        if $T \subset W$ then
            for each $M \in T$ do
                $\mathcal{V} \leftarrow \mathcal{V} \uplus \{\{M\}\}$
            end for
        end if
    end for
    return $\mathcal{V}$
end function
```

```
function GET_GPT($N$)
    $P$ = POS($N$)                    // Get Part-of-Speech for the node using Stanford Parser
    if $P$ == (NN | NNS | NNP | NNPS$) then
        return NP
    else if $P$ == (VBD | VBG | VBN | VBP | VBZ$) then
        return VB
    else
        return $P$
    end if
end function
```

4.4.4.1 P(f|e)

The model $P(f|e)$ with $f \in F$ and $e \in E$ encodes the probability of a frame given an event. Intuitively, it captures the suitableness of the frame f to verbalise the event e as observed from the training corpus and provides an estimate for selecting the best frame to verbalise a given event during test time. It is computed as follows :

$$P(f|e) = \frac{counts\left((f, e) \in \mathcal{C}_e\right) + 0.1}{\sum_{f'} \left(counts\left((f', e) \in \mathcal{C}_e\right) + 0.1\right)} \tag{4.1}$$

where $\mathcal{C}_e$ represents the collection of all frames extracted for the event e from the sentence corpus; $counts(f, e)$ is the number of times the frame f is observed for the event e in $\mathcal{C}_e$ and $counts(f', e)$ is the frequency of any frame f' observed for the event e in $\mathcal{C}_e$. $\mathcal{C}_e$ is directly obtained from the frame extraction procedure discussed in Section 4.4.3 above and the computation of $P(f|e)$ becomes straightforward.

4.4.4.2 P(f|r)

The model $P(f|r)$ with $f \in F$ and $r \in R$ encodes the probability of a frame given a role. Intuitively, its a measure of how appropriately a frame f qualifies for expressing a given semantic role r and it helps to rank the frames based on the given set of semantic roles during test time. It is computed as follows :

$$P(f|r) = \frac{counts\left((f, r) \in \mathcal{C}_r\right) + 0.1}{\sum_{f'} \left(counts\left((f', r) \in \mathcal{C}_r\right) + 0.1\right)} \tag{4.2}$$

where $\mathcal{C}_r$ represents the collection of all frames aligned to the role r; $counts(f, r)$ is the number of times the frame f is observed for the role r in $\mathcal{C}_r$ and $counts(f', r)$ is the frequency of any frame f' observed for the role r in $\mathcal{C}_r$.

The computation of $\mathcal{C}_r$ is bit involved and takes into account the global alignment of all the KBGen$^+$ roles with all the KBGen$^+$ event frames extracted from the sentence corpus. Using a toy example (Example 1) resembling the KBGen$^+$ dataset (triples are in prefix notation (event followed by entity) and separated by space) and sample sentences taken from the corpus, we explain the assumptions underlying such alignment and demonstrate the computation of $P(f|r)$ below.

Example 1

Event Descriptions :

A : instrument(Maintain,Electrogenic-Pump) object(Maintain,Membrane-Potential)

B : base(Release,Lysosome) destination(Release,Food-Vacuole) object(Release,Hydrolase)

C : agent(Create,Electrogenic-Pump) result(Create,Membrane-Potential)

Sentences :

A : *Mitochondria maintains a membrane potential.*

B : *Hypertrophic chondrocytes release hydrolases.*

Given the event descriptions and the sentences, our frame extraction procedure generates the frame nsubj,VB,dobj for the event maintains (from Sentence A) and release (from Sentence B). The dependency subtrees generating the frame for the corresponding events are shown in Figure 4.3 and 4.4 respectively.

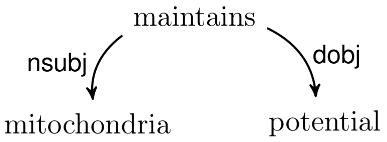


Figure 4.3: Dependency subtree : Maintain

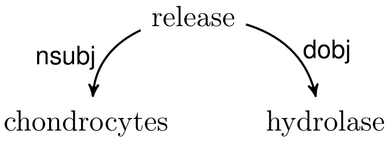


Figure 4.4: Dependency subtree : Release

From such dependency subtrees generating the frames for the events in the input, we target learning the frame-role alignment as follows. We assume that a event frame f given by a dependency subtree t can be aligned to any role r in the total input whenever t has some entity e as its dependent and the entity e is bound via the role r in any of the event descriptions of the input. This assumption is motivated by the goal of identifying all the semantic roles a given frame is likely to express in the context of the global input, independent of the particular event generating that frame.

In Figure 4.3, the dependency subtree verbalises the entity MEMBRANE-POTENTIAL present in the input since our lexicon provides the term "potential" as once of its verbalisation alternatives. It provides the frame nsubj,VB,dobj and based on the assumption just discussed, we look for the roles from the total input that can be aligned to this frame. In Event Description A, we have the role OBJECT and in Event Description C, we have the role RESULT that bind this entity. We therefore, align the roles OBJECT and RESULT to the frame nsubj,VB,dobj and indicate their frequency of occurrence as in Table 4.6 below.

	result	**object**
nsubj,VB,dobj	1	1

Table 4.6: $\mathcal{C}_r$ after processing Event MAINTAIN

In Figure 4.4, the dependency subtree also generates the same frame nsubj,VB,dobj but verbalises a different entity HYDROLASE present in the input. The entity HYDROLASE is bound via the role OBJECT in the Event Description B and we augment Table 4.6 to account this observation. Table 4.7 shows the increment in the frequency value for the role OBJECT since the frame happens to be the same.

	result	**object**
nsubj,VB,dobj	1	$1 + 1$

Table 4.7: $\mathcal{C}_r$ after processing Events MAINTAIN and RELEASE

In the toy Example 1 we present, none of the sentences verbalise the event CREATE from the Event Description C, yet we benefit from using the role information in it to build up $\mathcal{C}_r$ (as discussed in building Table 4.6). This means that our frame-role alignment strategy abstracts over the actual events generating the frames and profits from such correspondences learnt from the global KBGEN$^+$ dataset. By using the total sentences in the corpus and all the event descriptions in the KBGEN$^+$ dataset, our $\mathcal{C}_r$ expands to cover all the frames that can be aligned with all the roles present in the dataset; updating the frequency information each time.

4.4.4.3 P(d|r)

In contrast to the earlier two models which aim at choosing frames given an event, the $P(d|r)$ aims at learning the syntax/semantic linking for verbalising the selected frame. Intuitively, it provides the likelihood estimates for mapping a syntactic dependency d to a semantic role r and is used to choose which dependencies in the selected frame should map to which roles in the input event description during test time. It is computed as follows :

$$P(d|r) = \frac{counts\left((d,r) \in \mathcal{C}_d\right) + 0.1}{\sum_{d'} \left(counts\left((d',r) \in \mathcal{C}_d\right) + 0.1\right)} \tag{4.3}$$

where $\mathcal{C}_d$ represents the collection of all dependencies aligned to the role r; $counts(d, r)$ is the number of times the dependency d is observed for the role r in $\mathcal{C}_d$ and $counts(d', r)$ is the frequency of any dependency d' observed for the role r in $\mathcal{C}_d$.

To compute $\mathcal{C}_d$, we assume that a dependency relation d present in a dependency subtree t can be aligned to any role r in the total input whenever d binds some entity e in t and the entity e is bound via the role r in any of the event descriptions of the input. Contrasting to the $P(f|r)$ model, here, we align the roles of the input to the corresponding dependency relation in the frame rather than the frame itself.

Using the same toy example (Example 1) above, we demonstrate the computation of $\mathcal{C}_d$. As discussed earlier, the dependency subtree in Figure 4.3 verbalises the entity MEMBRANE-POTENTIAL which is bound via the roles OBJECT and RESULT in the input. Here, we are interested in aligning the dependency relation binding the entity MEMBRANE-POTENTIAL in the dependency subtree, i.e. the *dobj* relation, against those roles and therefore have the frequency information as shown in Table 4.8.

	result	**object**
dobj	1	1

Table 4.8: $\mathcal{C}_d$ after processing Event MAINTAIN

Similarly, in Figure 4.4, the dependency subtree verbalises the entity HYDROLASE which is bound via the role OBJECT in the input and by the dependency relation *dobj* in the tree. Updating the frequency information from this observation gives us Table 4.9.

	result	**object**
dobj	1	$1 + 1$

Table 4.9: $\mathcal{C}_d$ after processing Events MAINTAIN and RELEASE

As with the $P(f|r)$ model, the alignment abstracts over the events and profits from learning over the global KBGEN$^+$ dataset. And similar to $\mathcal{C}_r$ computation, the $\mathcal{C}_d$ expands to account for observations made from processing all the event descriptions in the KBGEN$^+$ dataset using the total sentences in the corpus.

For each of the three models $P(f|e)$, $P(f|r)$ and $P(d|r)$; we assume them to be multinomial with their maximum likelihood estimates determined by the labelled data, $\mathcal{C}_e$, $\mathcal{C}_r$ and $\mathcal{C}_d$ respectively. Also, we choose to model the conditional probabilities for frames and dependencies given events and roles, and not the other way around, because such models intuitively match the generation process at test time. Algorithm 2 outlines the steps we follow in building $\mathcal{C}_e$, $\mathcal{C}_r$ and $\mathcal{C}_d$.

Algorithm 2 Preparation of the corpora used to train our probabilistic models

```
  Input   KBGen$^+$
          Lexicons $\mathcal{L}_e$ for events and $\mathcal{L}_t$ for
              entities as described in Section 4.4.2
          Raw text corpus $\mathcal{T}$ with dependency
              trees as described in Section 4.4.3
 Output   Corpus (multiset) $\mathcal{C}_e$ for model $P(f|e)$
          Corpus (multiset) $\mathcal{C}_r$ for model $P(f|r)$
          Corpus (multiset) $\mathcal{C}_d$ for model $P(d|r)$

1. For every event $e \in$KBGen$^+$ let $lex(e)$ be all possible lexicalisations of $e$ taken
   from $\mathcal{L}_e$:
2. For every lexicalisation $l \in lex(e)$:
3. For every occurrence $e_t \in \mathcal{T}$ of $l$:
   (a) Extract the frame $f$ governed by $e_t$
   (b) Add the observation $f$ with label $e$ in the frame-event corpus:
           $\mathcal{C}_e \leftarrow \mathcal{C}_e \uplus \{\{(f, e)\}\}$
   (c) For every entity $w_t \in \mathcal{L}_t$ that is a dependent of $e_t$ with syntactic relation
       $d$, add every role $r$ associated with this entity in KBGen$^+$ to both role
       corpora:
           $\mathcal{C}_r \leftarrow \mathcal{C}_r \uplus \{\{(f, r)\}\}$
           $\mathcal{C}_d \leftarrow \mathcal{C}_d \uplus \{\{(d, r)\}\}$
```

4.4.5 Surface Realisation

The input to the surface realisation task are individual event descriptions of the KBGen$^+$ dataset. To verbalize an input event description, we first identify the event e and the set of roles $r_1 \ldots\ldots r_n$ it contains. We define the arity of an event as the count of distinct role types present in the event description in order to account for aggregation of repeated roles, if any. Thus if the input event description contains e.g., 2 object roles and an instrument role, its arity will be 2 rather than 3 and this accounts for the fact that the two object roles will be verbalised as a coordinated NP filling in a single dependency function rather than two distinct syntactic arguments in the generated sentence. Then, from all the frames present in $\mathcal{C}_e$ for this event, we select only those that have the same arity (this equals to the number of syntactic dependents in case of frames) as the input event. All such frames are candidate frames for our surface realisation task.

We consider two alternative scoring functions for scoring the candidate frames. We aim to identify the 5 best frames for each input and set n=5.

In the first model, we select the frame f which maximises the score (M1):

$$P(f|e) \times \prod_{i=1}^{n} P(f|r_i) \tag{M1}$$

For the second model, we want to have a scoring function that also takes into account the optimal mapping of syntactic dependents in the frame $(d_1, \ldots, d_n)$ to the roles $(r_1, \ldots, r_n)$ in the input for scoring the frame itself. Thus, we first define a function $(\hat{r}_1^f, \ldots, \hat{r}_n^f)$ which determines the best permutation for one-to-one mapping of $(d_1, \ldots, d_n)$ in a given frame f to $(r_1, \ldots, r_n)$ in the input (Equation 4.4) :

$$(\hat{r}_1^f, \ldots, \hat{r}_n^f) = \underset{(s_1,\ldots,s_n)\in\mathcal{P}(\{r_1,\ldots,r_n\})}{\text{argmax}} \prod_{i=1}^{n} P(d_i|s_i) \tag{4.4}$$

where $\mathcal{P}(\{r_1, \ldots, r_n\})$ is the set of all permutations of the roles[21].

and use the mappings proposed by Equation 4.4 to compute the scores in model (M2) :

$$P(f|e) \times \prod_{i=1}^{n} P(f|r_i) \times \prod_{i=1}^{n} P(d_i|\hat{r}_i^f) \tag{M2}$$

Note that both (M1) (and (M2)) can be viewed as a *product of experts* [Hinton, 2002] but with independently trained experts and without any normalization factor. It is thus not a probability, but this is fine because the normalization term does not impact the choice of the winning frame.

Both models (M1) and (M2) generate individual scores for the candidate frames. By ranking the scores, we identify the top n-best scoring frames and retain only them for generation purposes. To verbalise each of those n-best frames (both from (M1) and (M2)), we need to determine the mapping between the syntactic dependents it contains and the roles in the input event description for which this frame was

[21] Here, we assume the order of dependencies in f is fixed, and we permute the roles; this is of course equivalent to permuting the dependencies with fixed roles sequences.

selected. Again, Equation 4.4 comes to rescue. Then, the generation boils down to filling every dependency slot in the frame with the lexical entry of the corresponding role's bound entity in the input and preserving the head word verbalising the event. For repeated roles of the input, we aggregate their bound entities via the conjunction "and" and fill the corresponding dependency slot.

Let us consider an example to see the surface realisation task in practice, the syntax/semantic mapping in particular. Let us assume that the test input is an event description with the event Block participating via binary roles instrument and object to entities Plasma-membrane and Hydrophobic-Compound respectively :

```
(|Block| |instrument| |Plasma-Membrane|)
(|Block| |object| |Hydrophobic-Compound|)
```

Let us assume that the following frames (and many more) are observed for the event Block in $\mathcal{C}_e$:

nsubj,cop,NP,prep_of,rcmod	Frame 1
nsubj,VB,acomp,prep_to	Frame 2
nsubj,VB,dobj	Frame 3
VB,prep_of,prep_to	Frame 4
....	

As we can see, only the frames Frame 3 and Frame 4 from the above list serve as candidate frames for verbalising the input event description as they are the only ones with the matching arity (i.e. 2).

Further, let us assume that Frame 3 is the top scoring frame (either from (M1) or (M2)) and the we have the following $P(d|r)$ probabilities (in log10 scale) from the training phase :

	instrument	**object**
nsubj	-1.25	-0.99
dobj	-1.15	-0.76

Table 4.10: Sample $P(d|r)$

For verbalising the frame Frame 3, Equation 4.4 permits the scoring of two different syntax/semantic mapping combinations :

instrument → dobj & object → nsubj $= -1.15 + -0.99 = -2.14$
instrument → nsubj & object → dobj $= -1.25 + -0.76 = -2.01$

and chooses the second mapping as it scores a higher value. Therefore, we proceed to verbalise the frame Frame 3 such that the entity bound by the INSTRUMENT role in the input (i.e. the PLASMA-MEMBRANE entity) fills the *nsubj* dependency slot and the entity bound by the OBJECT role in the input (i.e. the HYDROPHOBIC-COMPOUND entity) fills the *dobj* slot in the frame. From the dependency subtree generating the frame Frame 3 , we have the information of the word form verbalising the event in the frame (i.e. the word *blocks*) and from the lexicon we have the word forms to express the entities (i.e. the word *Plasma membrane* and *Hydrophobic compounds* respectively). Thus the generated sentence is :

(18) *Plasma membrane blocks Hydrophobic compounds.*

The results obtained by verbalising the n-best frames given by models (M1 & M2) are separately stored and we present their analysis in Section 4.5.

4.5 Results and Evaluation

We evaluate our approach on the 336 event representations included in the KBGEN$^+$ dataset. As discussed in the preceding section, for each event representation in the input, we extract the 5 best natural language verbalisations according to the score provided by the models just discussed. To evaluate the generated sentences, we need to compare them against some reference sentences, which we create by manually editing the reference sentences provided by the KBGEN dataset. As discussed earlier, our KBGEN$^+$ dataset is the subset of the KBGEN dataset where only the relations forming an event description are retained. Thus, we create reference sentences for our KBGEN$^+$ dataset by retaining only the event description structures in the KBGEN sentences with minimal edits and careful manual analysis. In the remainder of this text, we shall refer to this set of reference sentences for our KBGEN$^+$ dataset as gold corups.

We evaluate the results both quantitatively (Automatic Evaluation) and qualitatively (Human Evaluation).

4.5.1 Automatic Evaluation

4.5.1.1 Coverage

We first consider coverage i.e., the proportion of input for which a verbalisation is produced. In total, we generate output for 321 event descriptions (95.53% of the total input).

For 3 input cases involving two distinct events (PHOTORESPIRATION and UNEQUAL-SHARING), there was no associated frame because none of the lexicalisations of the event could be found in the corpus. Covering such cases would involve a more sophisticated lexicalisation strategy for instance, the strategy used in [Trevisan, 2010], where the word forms are tokenized and pos-tagged before being mapped using hand-written rules to a lexicalisation.

For the other 12 input cases, generation fails because no frame of matching arity could be found. Several factors account for this failure and we discuss them in detail in Section 4.6 below.

4.5.1.2 Accuracy

Because the generated verbalisations are not learned from a parallel corpora, the generated sentences are often very different from the reference sentence. For instance, the generated sentence may contain a verb while in the reference sentence, the event is nominalised. Or the event might be verbalised by a transitive verb in the generated sentence but by a verb taking a prepositional object in the reference sentence (Eg: *A double bond holds together an oxygen and a carbon* vs *Carbon and oxygen are held together by double bond*). To automatically assess the quality of the generated sentences, we therefore do not use BLEU. Instead we measure the accuracy of role mapping and we complement this automatic metric with the human evaluation described in the next section.

Role mapping is assessed as follows. First, for each event description in the input, we record the mapping between the KB role of an argument in the event description and the syntactic dependency of the corresponding natural language argument in the gold sentence. For instance, given the event description shown in Section 4.4.3 for Sentence 10 (repeated below for convinience as Example 2), we record the syntax/semantics mapping *instrument:nsubj, object:dobj, base:prep-in.*

Example 2

```
(|Block| |instrument| |PC/EBP|)
(|Block| |object| |TNF-activation|)
(|Block| |base| |Myeloid-Cells|)
```

C/EBP beta blocked TNF activation in myeloid cells.

Accuracy is then the proportion of generated role:dependency mappings which are correct i.e., those that match the mappings in the gold sentence. Although this does not address the fact that the generated and the reference sentence may be very different, it provides some indication of whether the generated mappings are plausible. We thus report this accuracy for the 1-best and 5-best solutions provided by our models ((M1) and (M2)), to partly account for the variability in possible correct answers. We compare our results to two baselines created manually. The first baseline (BL-LING) is obtained using a default role/dependency assignment which is manually defined using linguistic introspection. The second (BL-GOLD) is a strong, informed baseline which has access to the frequency of the role/dependency mapping in the gold corpus. That is, this second baseline assigns to each role in the input event description, the syntactic dependency most frequently assigned to this role in the gold corpus. The default mapping used for BL-GOLD is as follows: *toward/prep_towards, site/prep_in, result/dobj, recipient/prep_to, raw_material/dobj, path/prep_through, origin/prep_from, object/dobj, instrument/nsubj, donor/prep_from, destination/prep_into, base/prep_in, away-from/prep_away_from, agent/nsubj.* The manually defined mapping used for BL-LING differs on three mappings namely *raw_material/prep-from, instrument-with,destination-to.*

On the 336 event descriptions (929 roles occurrences) contained in the dataset, we obtain the role mapping accuracies as shown in Table 4.11. For the baselines, we have only one proposition of role mapping, hence the results for 1-best accuracy only. However, for the models (M1) and (M2), we have 5 different propositions for role mappings, each coming from one of the 5-best generated sentences. Thus, we report the highest accuracy among the 5-best (the 5-best accuracy) and the one obtained from the topmost scoring output (the 1-best accuracy).

Scoring	5-best acc	1-best acc
BL-Ling		42%
BL-GOLD		49%
M1	48%	30%
M2	49%	31%
M2-BL-LING	57%	43%

Table 4.11: Role Mapping Accuracies

As expected, the difference between BL-LING and BL-GOLD shows that using information from the GOLD strongly improves accuracy.

While M1 and M2 do not improve on the baseline, an important drawback of these baselines is that they may map two or more roles in an event description to the same dependency (e.g., the roles RAW-MATERIAL and RESULT to the dependency *dobj*).

Worse, they may map a role to a dependency which is absent from the selected frame (if the dependency mapped onto by a role in the input does not exist in that frame). In contrast, our models (M1) and (M2) are linguistically more promising as they guarantee that each role is mapped to a distinct dependency relation. We therefore take advantage of both the linguistically inspired baseline (BL-LING) and the probabilistic approach by combining both into a model (M2-BL-LING) which simply replaces the mapping proposed by the M2 model by that proposed by the BL-LING baseline whenever the probability of the M2 model is below a given threshold[22]. Because it makes use of the strong prior information contained in the BL-LING baseline, it has a good accuracy.

4.5.2 Human Evaluation

For human evaluation, we take a sample of 264 inputs from the KBGen$^+$ dataset and evaluate the mappings of roles to syntax in the sentences generated from the (M2) model. The sample contains inputs with 1 to 2 roles (40%), 3 roles (30%) and more than 3 roles (30%). For each sampled input, we consider the 5 best outputs and manually grade the output as follows:

1. Correct: both the syntax/semantic linking and the lexicalisation of the event and entities is correct. An example of such output is shown below :

   Input :

   ```
   (|Radioactive-Treatment| |object| |Cancer|)
   (|Radioactive-Treatment| |instrument| |Radioactive-Isotope|)
   ```

   Generated Sentence : *Cancer is treated with radioactive isotope.*

2. Almost Corrrect: the lexicalisation of the event and entities is correct and the syntax/semantic linking of core semantic roles is correct. The core roles are the ones occuring more frequently in the KBGen$^+$ dataset, namely AGENT, BASE, OBJECT. An example of such output is shown below :

   Input :

   ```
   (|Share| |agent| |Nitrogen|)
   (|Share| |agent| |Hydrogen|)
   (|Share| |object| |Valence-Electron|)
   (|Share| |result| |Single-Bond|)
   ```

[22] We have empirically chosen a threshold that retains 40% of our model's outputs; this is the only threshold value that we have tried, and we have not tuned this threshold at all.

Generated Sentence : *Nitrogen and hydrogen share valence electron to single bond.*

3. Incorrect: all other cases. An example of such output is shown below :

   Input :

   ```
   (|Divide| |site| |Kinetochore-Microtubule|)
   (|Divide| |object| |Protein|)
   ```

   Generated Sentence : *Kinetochore microtubules share protein.*

Three judges independently graded the generated sentences for the 264 samples using the above criteria. The inter-annotator agreement, as measured with the Fleiss Kappa in a preliminary experiment in these conditions, was $\kappa = 0.76$ which is considered as "good agreement" in the literature. 29% of the ouput were found to be correct, 20% to be almost correct and 51% to be incorrect.

4.6 Discussion

A manual analysis of the results obtained exposes the limitations inherent to our approach and provides insights for improvements.

4.6.1 Events with no or little Training Data.

One of the causes of generation failure in our approach is that none of the sentences in the corpus describe the input event and therefore no frames can be extracted for that event. As presented earlier, this accounts for 3 cases in total but a more detailed analysis underlines the problem with our sentence corpus – for 17.6% of total events, less than 20 distinct frames could be extracted while 56.8% of events had more than 80 frames from the training phase. This highlights the fact that our sentence corpus is not a uniform dataset for learning and points at the data sparsity problem. Increasing the training corpus using sentences from additional sources could help to lessen the problem.

An alternative technique to address the lack of frames in the observed corpus would be to import the verb subcategorization information from existing wide coverage verb lexicon such VerbNet [Kipper *et al.*, 2000] and FrameNet [Baker *et al.*, 1998]. In this approach, one or more of the word forms available for events in our input could be used for matching against the entries in the verb lexicon and corresponding frames be assigned to our event. Furthermore, a predefined set of event frames could be manually added to the collection of observed frames by carefully analysing the input and this would cater to a default/better frame selection.

4.6.2 Frame Arity Mismatch

The second and the most frequent cause of generation failure in our approach is the mismatch in the event frame arity during training and testing phases. In this case, although one or more frames have been observed for an event during training, none of those frames match the arity of the event in the given test input and therefore cannot be used for generation. To address this shortcoming, event frames observed during training can be used for automatically generating frames with lower arity; for example by pruning one or more optional arguments in them such as the prepositional arguments, verbal modifiers and clausal complements. When augmented in this manner, the newer frames obtained for an event either improve the frequency distribution (if already observed from other sentences in the corpus) or remedy the lack of lower arity frame for the given event. It is however, not clear how one can address the needs for higher arity frames.

4.6.3 High-arity Events

Besides the coverage issue, several factors account for the low quality of the generated sentences. One major factor negatively affecting the quality of the output sentences is the number of roles contained in an event description. Unsurprisingly, the greater the number of roles, the lower the quality of output sentence. That is, for event descriptions with 3 or less roles, the number of correct output is higher (40%, 23%, 37% respectively for correct, almost correct and incorrect) as there are less possibilities to be considered. Further, the roles that are less frequent in the KBGen^{+} dataset often score lower probabilities (i.e., are more often incorrectly mapped to syntax) than roles which occur more frequently. Thus, the three most frequent roles (AGENT,OBJECT, BASE) have a 5-best role mapping accuracy that ranges from 43% to 77%, while most other roles have much lower accuracy. Again, this points out the data sparsity problem and could be improved by using either more data or a more sophisticated smoothing or learning strategy. However, linguistic factors are also at play here (described in Section 4.6.4 and 4.6.5 below).

4.6.4 Not all Semantic Roles are verbalised by thematic roles

First, some semantic roles are often verbalised as verbs rather than thematic roles in the gold corpus. For example, the reference sentence in the gold corpus for the input in Example 3 is shown in Sentence (19).

Example 3

```
(|Intracellular-Digestion44355| |object| |Polymer44401|)
(|Intracellular-Digestion44355| |object| |Solid-Substance44361|)
(|Intracellular-Digestion44355| |result| |Monomer44397|)
(|Intracellular-Digestion44355| |site| |Lysosome44357|)
```

Reference Sentence :

(19) Intracellular digestion of polymers and solid substances in the lysosome produces monomers.

As we can see, in this sentence, the event INTRACELLULAR-DIGESTION is verbalised as a noun (*Intracellular digestion*) and the role RESULT as a verb (*produces*). More generally, a role in the KB is not necessarily realised as a thematic role of the event. This is mostly true for events which are often verbalised in their nominalized form (e.g. the events INTRACELLULAR-DIGESTION, PHOTOSYNTHESIS, RESPIRATION etc.). Our approach inherently assumes that events are verbalised as verbs rather than nouns and is therefore not sufficient to address this issue.

Note that although we can extract NP rooted frames (in addition to the usual VB rooted frames, for example when using the lexical entry *digestion* for INTRACELLULAR-DIGESTION) for such events; the frames so obtained are not good for our task because they either lead to ungrammatical sentence generation or are extracted because of dependency parser error. Consider the scenarios in Example 4 and 5. Example 4 shows the NP rooted frame "amod,NP,prep_of,vmod_producing" extracted for the event INTRACELLULAR-DIGESTION from some sentence (Sentence (20)) in the training corpus. When this frame is used to generate from the input in Example 3, we obtain an ungrammatical sentence as shown in Sentence (21). Example 5, on the other hand, shows a NP rooted frame extracted for the event PHOTOSYNTHESIS arising from the Sentence (22) because of dependency parser error.

Example 4

(20) *Orlistat partially blocks intestinal digestion of fat, producing weight loss.*

Dependency subtree around the event INTRACELLULAR-DIGESTION using it's lexical entry *digestion* :

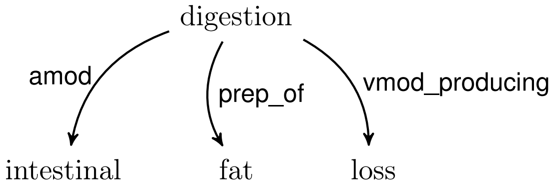


Figure 4.5: Dependency subtree : INTRACELLULAR-DIGESTION

Frame extracted : amod,NP,prep_of,vmod_producing

Generated Sentence for the Input in Example 3:

(21) *Lysosome digestion of polymer and solid substances producing monomer.*

Example 5

(22) *The cotyledons of the dicots function similar to leaves (i.e. perform photosynthesis).*

Dependency subtree around the event PHOTOSYNTHESIS using it's lexical entry *photosynthesis* :

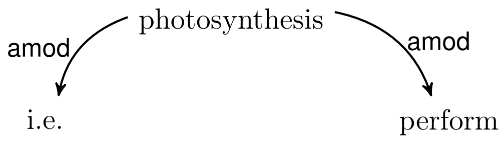


Figure 4.6: Dependency subtree resulting from parser error

4.6.5 Semantic Arguments of the Events verbalised as NP modifiers

Second, in some cases, an entity which is an argument of the event in the input is verbalised as prepositional modifiers of some other entity in the same event description rather than as an argument of the event. This is frequently the case for the entity related by the role BASE in the event description. For instance, the Sentence (23) shows the gold sentence for an input containing the entity EUKARYOTIC-CELL as a BASE argument. As we can see, in this case, the entity EUKARYOTIC-CELL is verbalised by a prepositional phrase modifying another entity LYSOSOME rather than as an argument of the event DIGEST.

(23) Lysosomal enzymes digest nucleic acids and proteins in the lysosome of eukaryotic cells.

Further study is needed to assess how the current approach can be extended to address both these issues.

4.7 Conclusion and Directions for Further Research

We have presented an approach for verbalising triples in ontologies which differs from previous work in that (i) it uses a non-parallel corpora and (ii) it focuses on n-ary relations and iii) on the issue of how to automatically map natural language and KB arguments. Our evaluation shows encouraging results and identifies three main open questions for further research. How best to deal with data sparsity to account for lack of frames and for event descriptions involving a high number of roles or roles that are infrequent? How to handle semantic roles that are verbalised as modifiers rather than as syntactic arguments? How to account for cases where KB roles are verbalised by verbs rather than by syntactic dependencies?

A more fine grained study of probabilistic models could be useful for obtaining better results. The probability models we have discussed here are based on distribution of two variables. For example $P(d|r)$ only takes into account two variables – the dependency d and role r variables. Instead, by conditioning the probabilities on additional variables, e.g. $P(d|r,e)$, we can obtain a different distribution that takes more variables into context. For this example, it means that the dependency d is ranked not just in terms of its likelihood to be associated to the given role r but also in the context that role r has been observed for the event e. The study of proper models that make for good generation output is worth exploring.

Further, the probability models used in this work for ranking the n-best frames extracted for event descriptions could be augmented with vector based approaches. In particular, some similarity scoring function like the one implemented in [Cheung and Penn, 2014] using vector space distribution of words could be used to compute the similarity of the arguments in a given input event description with the arguments present in the subtree from which the frame is extracted for that event description.

These scores obtained for the respectived frames could, in turn, be combined with the probablistic scores we obtain; in a log-linear fashion as described in [Lu and Ng, 2011] to obtain a final ranking of the extracted frames.

From a practical perspective, an evaluation of our approach across different KB domains remains an open challenge. Spanning it across different KB domains will call for domain-specific practical considerations. Firstly, the availability of textual corpora in the given domain is a stringent requirement. Also, the availability of appropriate lexicon and domain-specific vocabularies will impact the coverage and quality of generated outputs. On a positive note, however, the approach discussed here is generic and can therefore be expected to scale up to different KB domains; provided such practical constraints are adequately met.

Finally, our work could be extended to account for generation of multi-sentences text for describing several related event descriptions at once. This would involve identifying multiple event descriptions from the input that describe a given event in relation to varying entities or different events that can be correlated in domain-specific context. Furthermore, in order to achieve a coherent discourse level text, the mutli-sentence text needs to introduce appropriate discourse markers rather than simply group single setences obtained from individual event descriptions. This calls for design principles spanning higher level NLG tasks (Content Planning and Micro Planning) in addition to the Surface Realisation task described in our work.

Chapter 5

Conclusion

Contents

5.1 Summary

We have presented two different approaches to Surface Realisation from Knowledge Bases. We explored this task in the context of verbalising RDF triples in ontologies and proposed novel methods for doing so. We started with an introduction to the Surface Realisation task, identified the challenges integral to it and outlined practical reasons that motivate the study of surface realisation. We then discussed the heterogeneous data formats/sources that form input to surface realisation in context of different applications and presented a comprehensive survey of existing approaches catering to the different input types. Following this, we focused our study of surface realisation particularly on the task of verbalising data in Knowledge Bases (ontologies) and proposed two novel approaches for doing so in Chapter 3 and 4 respectively.

In Chapter 3, we proposed a supervised approach to surface realisation from KB data. There, we presented our automatic approach for learning symbolic grammar from parallel data/text corpus for mapping RDF triples in ontologies to text in sentences. The novelty in this approach lied in i) proposing a new grammar induction procedure in the context of mapping KB data to text in sentences and ii) building a grammar which encodes both the syntactic and semantic information. We induced the grammar conforming to the linguistic principles of TAG and demonstrated that our approach allows for inducing a grammar which is compact, can feasibly be adapted to cover unseen cases and restricts overgeneration.

In Chapter 4, we presented the weakly supervised approach which targeted learning of verbalisation patterns for ontology triples from non-parallel text corpora. We developed different probabilistic models accounting for the choice of syntactic frames and their mapping of semantic arguments while verbalising event descriptions in ontologies. Learning verbalisation patterns from non-parallel corpora and building probabilistic models to induce plausible syntactic/semantic argument linking information constitute the novel aspects of this work. We presented examples of output sentences obtained by utilizing such models; analysed them and identified the issues relevant to this approach.

As already discussed in Chapter 2 (Section 2.3), ontologies represent an increasily popular choice of knowledge representation and sharing format across the web. In recent years, we have seen a huge surge in the use of domain-ontologies for automated knowledge sharing and having a natural language based access would significantly aid to their human consumption and growth. In this context, many recent works in surface realisation have focused on generation from these data sources. In this thesis, we contributed to this growing line of research by proposing two novel approaches. Both the approaches we presented here are generic and are independent of the ontology domain. Our approaches are suited to the verbalisation of multiple triples at once and are based on learning of generation resource from parallel and non-parallel texts, thus avoiding the drawbacks of handcrafted authoring as prevalent in many existing systems.

For each of our approaches, we have presented a detailed description of the procedure involved using examples, wherever appropriate. We have shown the results obtained and performed their analysis following automatic and human evaluations. We identified possible problem cases and provided linguistic insights into the causes for failures. In the context of addressing those shortcomings and from a broader perspective of a long term research, we envision several research avenues for our work as outlined in the following section.

5.2 Directions for Future Research

- Automatic Data-to-Text Alignment : Both the approaches that we have presented in this thesis are based on learning from the sentences in a given corpus (either parallel or non-parallel) and integral to this approach is the Data-to-Text Alignment model which essentially relies on the supplied lexicon for mapping of KB symbols in input to word forms in the sentences. While we used a lexicon for such alignment, it would be interesting to explore alternative methods which propose learning such alignments from the corpus sentences themselves. [Liang *et al.*, 2009], for example, propose a generative model which learns the most probable word sequence in a sentence for the given data value in the input database. They observe the distribution of words given the data type (integer, string and categorical fields) and the data value itself over the whole corpus and use probability scores to determine the most likely sequence of words expressing the data value. In our context, following similar approach would involve building a generative model which reflects the distribution of words in the corpus given the variable types (event, entity and property-values) and their values (e.g. gated channel) and provides the most likely word sequence for alignment. [Walter *et al.*, 2013] propose a semi-automatic method for creating lexical entries of concepts and relations present in ontologies. They use a domain corpus; extract the syntactic structures which bind the arugments of the given relation and assign the frequent syntactic structures as the lexical item for that relation. Following this approach, we could, for example, learn the lexical items for event variables (they capture the relationships between entity variables in the input) from the corpus and use those lexical items to guide our Data-to-Text Alignment strategy. Effectively, such approaches would enable us to make build a system independent of hand-written lexical resources.

- Improving the Quality of Generated Sentences : For improving the quality of generated sentences, we envision a process in which human feedback is incorporated in learning the verbalisation patterns of semantic relations in the input. Such method has been proposed, for example, in [Wang *et al.*, 2015] in which the authors first generate a cannonical verbalisation of logical expressions and request crowdsourced human edits for obtaining grammatical paraphrases. They show that the process is feasible and time efficient. In our context, we could present a small set of generated sentences (along with their KB input) and use similar crowdsourcing approach for obtaining alternative/better verbalisations of the corresponding inputs. The resulting sentences so obtained for each input can be expected to represent natural human utterances set in the context of verbalising KB data rather than the generic nature of sentences in domain corpora which are intended for general purposes. Especially with our weakly supervised approach, where we use multiple non-parallel corpora, learning the syntax/semantic mapping from such human reviewed sentences and analysing its effects on the quality of generated sentences is worth exploring.

- Alternative Probabilistic Models for the Weakly supervised Approach : In the weakly supervised approach, we observed the distribution of frames in the corpus and proposed different probability models to account for the choice of best frames and the mapping of semantic relations in the input to syntactic dependencies in those frames. Our models, however, are representative of only a few of the several probability models that can be built from the same distribution. For this work, we based our decisions on proposing models which intuitively reflected the generation process and were conceptually simple. A study into alternative probability models that can be derived from the same distribution would allow for comparing the quality of current models. Furthermore, when combining those models for the surface realisation task, we assumed them to be independent of each other and used a simple scoring function. Instead, using some Markovian conditioning for chaining of the probability models, for example, to consider the order of roles in the event description would allow for different propositions of frames and syntax/semantics mapping than the ones we have currently adopted in this work. A linear combination strategy of our probability models using weights learnt automatically (designing features to represent events, entities, relations and modeling the conditional dependencies of these variables in terms of their feature values; the approach followed in Conditional Random Fields [Lafferty, 2001]) or set after a careful emperical analysis presents another possibility of research. Exploring such alternative forms of probability models and their combinations will help in investigating the best verbalisation scheme.

- Different Data Types and Application Domains : In Section 2.2.1, we discussed the three different data types that make the input to surface realisation tasks, namely the databases, linguistic structures and logical forms. In this thesis, we pursued the surface realisation task for the specific input type coming from KBs which represent the "logical forms" category. As discussed earlier, the syntax, structure and content of each of these input types vary widely. However, our approaches in this thesis have been formulated on the basic notion of capturing relations existing between data items and each of these input types essentially express some relations existing among their data items, although in a very different way. It is, therefore, only logical to assume that our approaches could be adapted to fit the syntactic structures in these different input types following a careful anaylsis of each input type. More importantly, such approaches could benefit from the corpus based learning strategies we have pursued in this thesis. Following the data/text alignment strategy that we used in our works, we could target corpus based learning from datasets from different application domains, such as the TACoS corpus [Regneri *et al.*, 2013] which provides a timestamp annotations of video data. Adapting our approach to different data types and application domains would help to study the generality and scalability of our approach.

- Integrated NLG Platform : In this thesis, we focused our research objectives specifically to the surface realisation task and studied it in greater detail. It would be interesting to study this work in integration with other phases of a typical full-scale NLG systems, i.e. the Content Planning and the Micro Planning tasks. Content Planning would help in automatically determining the interesting/coherent fragments of KB data and organising them into appropriate discourse plans, possibly allowing for generation of multi-sentence texts by grouping the relevant fragments of data. Studies in Micro Planning would help for automatic lexicalisation, integration of referring expression and aggregation of similar content units. When integrated with our surface realisation approach, this would allow for an end-to-end generation system with fully autonomous selection, organisation and verbalisation of fragments of knowledge bases relevant to the changing user goals in a dynamic NLG application.

Appendices

Appendix A

Results from Supervised Approach

In this chapter, we present some representative examples of output obtained from our supervised approach. In Chapter 3, we have already shown examples of sentences generated while verbalising different relation types in the input (Event-to-Entity, Entity-to-Event, Event-to-Event, Entity-to-Entity and Entity-to-Property Values). Here, instead, we present sample outputs reflecting the qualitative nature of generated sentences – Correct vs. Incorrect.

We present the sample cases with their input, generated sentence and the corresponding reference sentence. Wherever appropriate, we remark on the cause of difference between the generated and the reference sentence.

Correct Result : Exact Match

```
:TRIPLES (
  (|Transfer22329| |object| |Electron22336|)
  (|Transfer22329| |donor| |NADH22341|)
  (|Transfer22329| |result| |FADH222344|)
  (|Transfer22329| |agent| |Electron-Shuttle-System22339|)
  (|Electron-Shuttle-System22339| |has-function| |Transfer22329|))
:INSTANCE-TYPES (
  (|Electron22336| |instance-of| |Electron|)
  (|NADH22341| |instance-of| |NADH|)
  (|FADH222344| |instance-of| |FADH|)
  (|Electron-Shuttle-System22339| |instance-of| |Electron-Shuttle-System|)
  (|Transfer22329| |instance-of| |Transfer|))
:ROOT-TYPES (
  (|Transfer22329| |instance-of| |Event|)
  (|Electron22336| |instance-of| |Entity|)
  (|NADH22341| |instance-of| |Entity|)
  (|FADH222344| |instance-of| |Entity|)
  (|Electron-Shuttle-System22339| |instance-of| |Entity|))
```

Generated Sentence :

The function of an electron shuttle system is to transfer an electron from nadh to fadh.

Reference Sentence :

The function of an electron shuttle system is to transfer an electron from nadh to fadh.

Correct Result : Non-Exact Match

```
:TRIPLES (
   (|Cellulose-Digestion12782| |object| |Cellulose12780|)
   (|Cellulose-Digestion12782| |raw-material| |Water-Molecule12783|)
   (|Cellulose-Digestion12782| |base| |Cell12789|)
   (|Cellulose-Digestion12782| |agent| |Cellulase12781|)
   (|Cellulase12781| |has-function| |Cellulose-Digestion12782|))
:INSTANCE-TYPES (
   (|Cellulose12780| |instance-of| |Cellulose|)
   (|Water-Molecule12783| |instance-of| |Water-Molecule|)
   (|Cell12789| |instance-of| |Cell|)
   (|Cellulase12781| |instance-of| |Cellulase|)
   (|Cellulose-Digestion12782| |instance-of| |Cellulose-Digestion|)
   (|Water-Molecule| |subclasses| |Chemical-Entity|))
:ROOT-TYPES (
   (|Cellulose-Digestion12782| |instance-of| |Event|)
   (|Cellulose12780| |instance-of| |Entity|)
   (|Water-Molecule12783| |instance-of| |Entity|)
   (|Cell12789| |instance-of| |Entity|)
   (|Cellulase12781| |instance-of| |Entity|))
```

Generated Sentence :

The function of cellulase is to digest cellulose in a cell, using water molecule.

Reference Sentence :

The function of cellulase is to digest cellulose in a cell, using a water molecule.

Remarks : The verbalisation for the entity variable `|Water-Molecule12783|` *was obtained during grammar adaptation by creating a default NP tree with its lexical information "water molecule"; thereby missing the indefinite determiner "a".*

Correct Result : Non-Exact Match

```
:TRIPLES (
   (|Diffusion-Of-Anion19310| |object| |Anion19306|)
   (|Diffusion-Of-Anion19310| |recipient| |Extra-Cellular-Matrix19307|)
   (|Diffusion-Of-Anion19310| |raw-material| |Membrane-Potential19309|)
   (|Membrane-Potential19309| |has-function| |Diffusion-Of-Anion19310|))
```

```
:INSTANCE-TYPES (
   (|Anion19306| |instance-of| |Anion|)
   (|Extra-Cellular-Matrix19307| |instance-of| |Extra-Cellular-Matrix|)
   (|Membrane-Potential19309| |instance-of| |Concentration-Gradient|)
   (|Diffusion-Of-Anion19310| |instance-of| |Diffusion-Of-Anion|))
:ROOT-TYPES (
   (|Diffusion-Of-Anion19310| |instance-of| |Event|)
   (|Anion19306| |instance-of| |Entity|)
   (|Extra-Cellular-Matrix19307| |instance-of| |Entity|)
   (|Membrane-Potential19309| |instance-of| |Entity|)))
```

Generated Sentence :

A function of concentration gradient is to provide energy for the diffusion of anions to extra cellular matrix.

Reference Sentence :

A function of a concentration gradient is to provide energy for the diffusion of anions and particles to the extra cellular matrix.

Remarks : The provided reference sentence is not an exact verbalisation of the input because it contains extra information "`particles`" *than that provided in the triples section of the input.*

Correct Result : Non-Exact Match

```
:TRIPLES (
   (|Photosynthesis2401| |raw-material| |Carbon-Dioxide2376|)
   (|Photosynthesis2401| |result| |Oxygen-Molecule2379|)
   (|Cellular-Respiration2386| |result| |Carbon-Dioxide2376|))
:INSTANCE-TYPES (
   (|Photosynthesis2401| |instance-of| |Photosynthesis|)
   (|Oxygen-Molecule2379| |instance-of| |Oxygen-Molecule|)
   (|Cellular-Respiration2386| |instance-of| |Cellular-Respiration|)
   (|Carbon-Dioxide2376| |instance-of| |Carbon-Dioxide|))
:ROOT-TYPES (
   (|Photosynthesis2401| |instance-of| |Event|)
   (|Carbon-Dioxide2376| |instance-of| |Entity|)
   (|Oxygen-Molecule2379| |instance-of| |Entity|)
   (|Cellular-Respiration2386| |instance-of| |Event|))
```

Generated Sentence :

Photosynthesis uses carbon dioxide and produces oxygen molecule.

Reference Sentence :

Cellular respiration produces carbon dioxide, which is used by photosynthesis to produce oxygen molecules.

Remarks : This is a case of partial generation; the entity `Carbon-Dioxide2376` *is shared across* *different event variables in the input.*

Correct Result : Non-Exact Match

```
:TRIPLES (
   (|Emit41865| |object| |Subatomic-Particle41867|)
   (|Emit41865| |base| |Atomic-Nucleus41869|)
   (|Radioactive-Isotope41868| |has-region| |Atomic-Nucleus41869|)
   (|Radioactive-Treatment41863| |object| |Cancer41864|)
   (|Radioactive-Treatment41863| |instrument| |Radioactive-Isotope41868|))
:INSTANCE-TYPES (
   (|Subatomic-Particle41867| |instance-of| |Subatomic-Particle|)
   (|Emit41865| |instance-of| |Emit|)
   (|Atomic-Nucleus41869| |instance-of| |Atomic-Nucleus|)
   (|Cancer41864| |instance-of| |Cancer|)
   (|Radioactive-Treatment41863| |instance-of| |Radioactive-Treatment|)
   (|Radioactive-Isotope41868| |instance-of| |Radioactive-Isotope|))
:ROOT-TYPES (
   (|Emit41865| |instance-of| |Event|)
   (|Subatomic-Particle41867| |instance-of| |Entity|)
   (|Atomic-Nucleus41869| |instance-of| |Entity|)
   (|Radioactive-Isotope41868| |instance-of| |Entity|)
   (|Radioactive-Treatment41863| |instance-of| |Event|)
   (|Cancer41864| |instance-of| |Entity|))
```

Generated Sentence :

Cancer is treated with subatomic particles emitted from atomic nucleus of radioactive isotope.

Reference Sentence :

A radioactive isotope whose nucleus emits subatomic particles is used in radiation treatment for cancer.

Remarks : The generated sentence is a correct paraphrase of the reference sentence; obtained after adaptation of different subtrees (learnt from different training scenarios) for this input.

Incorrect Result

```
:TRIPLES (
   (|Diffusion-Of-Anion19310| |object| |Anion19306|)
   (|Diffusion-Of-Anion19310| |agent| |Transport-Protein19295|)
   (|Diffusion-Of-Anion19310| |donor| |Cytoplasm19308|)
   (|Diffusion-Of-Anion19310| |raw-material| |Membrane-Potential19309|)
   (|Membrane-Potential19309| |has-function| |Diffusion-Of-Anion19310|))
:INSTANCE-TYPES (
   (|Anion19306| |instance-of| |Anion|)
   (|Transport-Protein19295| |instance-of| |Transport-Protein|)
   (|Cytoplasm19308| |instance-of| |Cytoplasm|)
   (|Membrane-Potential19309| |instance-of| |Concentration-Gradient|)
   (|Diffusion-Of-Anion19310| |instance-of| |Diffusion-Of-Anion|))
:ROOT-TYPES (
   (|Diffusion-Of-Anion19310| |instance-of| |Event|)
   (|Anion19306| |instance-of| |Entity|)
   (|Transport-Protein19295| |instance-of| |Entity|)
   (|Cytoplasm19308| |instance-of| |Entity|)
   (|Membrane-Potential19309| |instance-of| |Entity|))
```

Generated Sentence :

A function of concentration gradient is to provide energy for the diffusion of anions from cytoplasm by transport proteins.

Reference Sentence :

A function of a concentration gradient is to provide energy for diffusion of anions from the cytoplasm using a transport protein.

Remarks : The language model ranks an alternative verbalisation pattern for the relation `agent` *as the highest scoring output.*

Incorrect Result

```
:TRIPLES (
   (|Sieve-Tube-Element37616| |has-part| |Ion-Channel37606|)
   (|Sieve-Tube-Element37616| |has-part| |Nucleus37644|)
   (|Sieve-Tube-Element37616| |has-part| |Cytoplasm37645|)
   (|Loading-Of-Sugar37791| |destination| |Sieve-Tube-Element37616|)
   (|Loading-Of-Sugar37791| |object| |Sugar37615|))
:INSTANCE-TYPES (
   (|Ion-Channel37606| |instance-of| |Ion-Channel|)
   (|Nucleus37644| |instance-of| |Nucleus|)
   (|Cytoplasm37645| |instance-of| |Cytoplasm|)
   (|Sieve-Tube-Element37616| |instance-of| |Sieve-Tube-Element|)
   (|Loading-Of-Sugar37791| |instance-of| |Loading-Of-Sugar|)
   (|Sugar37615| |instance-of| |Sugar|))
:ROOT-TYPES (
   (|Sieve-Tube-Element37616| |instance-of| |Entity|)
   (|Ion-Channel37606| |instance-of| |Entity|)
   (|Nucleus37644| |instance-of| |Entity|)
   (|Cytoplasm37645| |instance-of| |Entity|)
   (|Loading-Of-Sugar37791| |instance-of| |Event|)
   (|Sugar37615| |instance-of| |Entity|))
```

Generated Sentence :

Sugar is loaded into sieve tube element containing the cytoplasm containing an ion channel containing the nucleus.

Reference Sentence :

Sugar is loaded into the sieve tube element, which consists of an ion channel, a nucleus and a cytoplasm.

Remarks : The repeating relation `has-part` *is separately verbalised for each dependent entity; leading to ungrammatical sentence generation.*

Incorrect Result

```
:TRIPLES (
   (|Constant-Motion10010| |object| |Molecule10022|)
   (|Constant-Motion10010| |speed| |Speed-Value10014|)
   (|Speed-Value10014| |directly-proportional| |Quantity-Value10013|)
   (|Molecule10022| |possesses| |Kinetic-Energy10012|)
   (|Kinetic-Energy10012| |quantity| |Quantity-Value10013|))
:INSTANCE-TYPES (
   (|Constant-Motion10010| |instance-of| |Constant-Motion|)
   (|Speed-Value10014| |instance-of| |Speed-Value|)
   (|Molecule10022| |instance-of| |Molecule|)
   (|Kinetic-Energy10012| |instance-of| |Kinetic-Energy|)
   (|Quantity-Value10013| |instance-of| |Quantity-Value|))
:ROOT-TYPES (
   (|Constant-Motion10010| |instance-of| |Event|)
   (|Molecule10022| |instance-of| |Entity|)
   (|Speed-Value10014| |instance-of| |Property-Value|)
   (|Quantity-Value10013| |instance-of| |Property-Value|)
   (|Kinetic-Energy10012| |instance-of| |Entity|))
```

Generated Sentence :

Molecule has kinetic energy.

Reference Sentence :

The speed of a moving molecule is directly proportional to the kinetic energy of the molecule.

Remarks : This is again a case of partial generation, but now the generated sentence is incorrect.

Appendix B

Results from Weakly Supervised Approach

In Chapter 4, we set the human evaluation criteria for evaluation of sentences generated from our weakly supervised approach in terms of “Correct”, “Almost Correct” and “Incorrect” results. The “Correct” results are the ones for which both the syntax/semantic linking and lexicalisation of event and entity variables are correct. For “Almost Correct” results, the lexicalisation of the event and entity variables is correct and the syntax/semantic mapping of the core roles (`agent`, `base` and `object`) is correct. The “Incorrect” results comprise all other instances.

Here, we present sample results from our weakly supervised approach conforming to those criteria. We further categorize the results in terms of the number of roles they verbalise (2 roles, 3 roles and higher than 3 roles) and for each result, we present its input and the corresponding reference sentence. Similar to the analysis of results from the supervised approach, we remark on the cause of difference between the generated and the reference sentence, wherever appropriate.

Correct : 2 roles

```
:TRIPLES (
   (|Transfer-Of-Malate8384| |object| |Malate8420|)
   (|Transfer-Of-Malate8384| |destination| |Bundle-Sheath-Cell8422|))
:INSTANCE-TYPES (
   (|Malate8420| |instance-of| |Malate|)
   (|Bundle-Sheath-Cell8422| |instance-of| |Bundle-Sheath-Cell|)
   (|Transfer-Of-Malate8384| |instance-of| |Transfer-Of-Malate|))
:ROOT-TYPES (
   (|Transfer-Of-Malate8384| |instance-of| |Event|)
   (|Malate8420| |instance-of| |Entity|)
   (|Bundle-Sheath-Cell8422| |instance-of| |Entity|))
```

Generated Sentence :

Malate is transferred to bundle sheath cell.

Reference Sentence :

Malate is transferred to a bundle sheath cell.

Correct : 3 roles

```
:TRIPLES (
   (|Cotransport15508| |subevent| |Active-Transport-Using-ATP15491|)
   (|Cotransport15508| |base| |Plant-Cell15472|)
   (|Cotransport15508| |agent| |S-H-ion-Cotransporter569|)
   (|Cotransport15508| |object| |Sucrose15460|))
:INSTANCE-TYPES (
   (|Active-Transport-Using-ATP15491| |instance-of| |Active-Transport-Using-ATP|)
   (|Plant-Cell15472| |instance-of| |Plant-Cell|)
   (|Cotransport15508| |instance-of| |Cotransport|)
   (|S-H-ion-Cotransporter569| |instance-of| |Sucrose-Hydrogen-ion-Cotransporter|)
   (|Sucrose15460| |instance-of| |Sucrose|))
:ROOT-TYPES (
   (|Active-Transport-Using-ATP15491| |instance-of| |Event|)
   (|Cotransport15508| |instance-of| |Event|)
   (|Plant-Cell15472| |instance-of| |Entity|)
   (|Sucrose15460| |instance-of| |Entity|)
   (|S-H-ion-Cotransporter569| |instance-of| |Entity|))
```

Generated Sentence :

Sucrose hydrogen ion cotransporter transports sucrose in plant cell.

Reference Sentence :

Sucrose-hydrogen ion cotransporter transports sucrose in a plant cell.

Almost Correct : 2 roles

```
:TRIPLES (
   (|Active-Transport24644| |object| |Hydrogen-Ion24643|)
   (|Active-Transport24644| |path| |Inner-Membrane24819|))
:INSTANCE-TYPES (
   (|Hydrogen-Ion24643| |instance-of| |Hydrogen-Ion|)
   (|Inner-Membrane24819| |instance-of| |Inner-Membrane|)
   (|Active-Transport24644| |instance-of| |Active-Transport|))
:ROOT-TYPES (
   (|Active-Transport24644| |instance-of| |Event|)
   (|Hydrogen-Ion24643| |instance-of| |Entity|)
   (|Inner-Membrane24819| |instance-of| |Entity|))
```

Generated Sentence :

Hydrogen ions are transported from inner membrane.

Reference Sentence :

Hydrogen ions are transported through an inner membrane.

Remarks : The event and the entity variables are lexicalised correctly but the subcategorisation frame selected for event variable doesn't offer the possibility of verbalising the semantic role “`path`” *via correct syntactic expression.*

Almost Correct : more than 3 roles

```
:TRIPLES (
   (|Release-Of-Calcium-Ion64543| |object| |Calcium-Ion64542|)
   (|Release-Of-Calcium-Ion64543| |raw-material| |Chemical-Energy64541|)
   (|Release-Of-Calcium-Ion64543| |agent| |Gated-Channel64540|)
   (|Release-Of-Calcium-Ion64543| |raw-material| |Membrane-Potential64538|))
:INSTANCE-TYPES (
   (|Calcium-Ion64542| |instance-of| |Calcium-Ion|)
   (|Chemical-Energy64541| |instance-of| |Chemical-Energy|)
   (|Gated-Channel64540| |instance-of| |Gated-Channel|)
   (|Membrane-Potential64538| |instance-of| |Concentration-Gradient|)
   (|Release-Of-Calcium-Ion64543| |instance-of| |Release-Of-Calcium-Ion|))
:ROOT-TYPES (
   (|Release-Of-Calcium-Ion64543| |instance-of| |Event|)
   (|Calcium-Ion64542| |instance-of| |Entity|)
   (|Chemical-Energy64541| |instance-of| |Entity|)
   (|Gated-Channel64540| |instance-of| |Entity|)
   (|Membrane-Potential64538| |instance-of| |Entity|))
```

Generated Sentence :

Gated channel releases calcium ion to chemical energy and concentration gradient.

Reference Sentence :

A gated channel releases calcium ions using chemical energy and a concentration gradient.

Remarks : The core semantic roles “`object`” and “`agent`” *are mapped to correct syntactic roles and with aggregation of similar roles but the syntax/semantic mapping for the role* “`raw-material`” is not correct.

Incorrect : 2 roles

```
:TRIPLES (
   (|Leave69963| |object| |Water-Molecule69961|)
   (|Leave69963| |destination| |Atmosphere69967|))
:INSTANCE-TYPES (
   (|Water-Molecule69961| |instance-of| |Water-Molecule|)
   (|Atmosphere69967| |instance-of| |Atmosphere|)
   (|Leave69963| |instance-of| |Leave|))
:ROOT-TYPES (
   (|Leave69963| |instance-of| |Event|)
   (|Water-Molecule69961| |instance-of| |Entity|)
   (|Atmosphere69967| |instance-of| |Entity|))
```

Generated Sentence :

Water molecules withdraw from atmosphere.

Reference Sentence :

Water molecules leave into the atmosphere.

Remarks : The subcategorisation frame obtained for the event variable “`leave`” *is based on its synonymous lexical entry* “`withdraw`” *which is not a correct verbalisation of the event in the context of specified input.*

Incorrect : 3 roles

```
:TRIPLES (
   (|Decomposition15146| |result| |Inorganic-Molecule15149|)
   (|Decomposition15146| |agent| |Fungus15140|)
   (|Decomposition15146| |object| |Detritus15145|))
:INSTANCE-TYPES (
   (|Inorganic-Molecule15149| |instance-of| |Inorganic-Molecule|)
   (|Decomposition15146| |instance-of| |Decomposition|)
   (|Detritus15145| |instance-of| |Detritus|)
   (|Fungus15140| |instance-of| |Fungus|))
:ROOT-TYPES (
   (|Decomposition15146| |instance-of| |Event|)
   (|Inorganic-Molecule15149| |instance-of| |Entity|)
   (|Fungus15140| |instance-of| |Entity|)
   (|Detritus15145| |instance-of| |Entity|))
```

Generated Sentence :

Detritus decomposes inorganic molecule into fungus.

Reference Sentence :

Fungus decomposes detritus into inorganic molecules.

Remarks : The frame chosen for verbalisation of the input event description is correct but the syntax/semantic mapping for none of the roles is correct.

Incorrect : more than 3 roles

```
:TRIPLES (
   (|Attach23994| |instrument| |Hydrogen-Bond24033|)
   (|Attach23994| |result| |Enzyme-Substrate-Complex23993|)
   (|Attach23994| |object| |Chemical-Entity24036|)
   (|Attach23994| |base| |Protein-Enzyme24003|)
   (|Attach23994| |site| |Active-Site24027|))
:INSTANCE-TYPES (
   (|Hydrogen-Bond24033| |instance-of| |Hydrogen-Bond|)
   (|Enzyme-Substrate-Complex23993| |instance-of| |Enzyme-Substrate-Complex|)
   (|Chemical-Entity24036| |instance-of| |Chemical-Entity|)
   (|Attach23994| |instance-of| |Attach|)
   (|Protein-Enzyme24003| |instance-of| |Protein-Enzyme|)
   (|Active-Site24027| |instance-of| |Active-Site|))
:ROOT-TYPES (
   (|Attach23994| |instance-of| |Event|)
   (|Hydrogen-Bond24033| |instance-of| |Entity|)
   (|Enzyme-Substrate-Complex23993| |instance-of| |Entity|)
   (|Chemical-Entity24036| |instance-of| |Entity|)
   (|Protein-Enzyme24003| |instance-of| |Entity|)
   (|Active-Site24027| |instance-of| |Entity|))
```

Generated Sentence :

In protein enzyme, enzyme substrate complex are attached along hydrogen bond to active site by chemical entity.

Reference Sentence :

An enzyme substrate complex is formed when a chemical attaches to the active site of a protein enzyme with a hydrogen bond.

Remarks : The semantic argument “`Active-Site24027`” *is a modifier of another argument* “`Protein-Enzyme24003`” *of the event rather than as a thematic argument of the event itself. Also, the semantic relation* “`result`” *is not realised as a thematic role of the event in the reference sentence.*

Bibliography

[Aguado *et al.*, 1998] G. Aguado, A. Bañón, J. Bateman, S. Bernardos, M. Fernández, A. Gómez-Pérez, E. Nieto, A. Olalla, R. Plaza, and A. Sánchez. ONTOGENERATION: Reusing Domain and Linguistic Ontologies for Spanish Text Generation. In *Workshop on Applications of Ontologies and Problem Solving Methods, ECAI*, volume 98, 1998.

[Amoia *et al.*, 2012] Marilisa Amoia, Treveur Bretaudiere, Alexandre Denis, C. Gardent, and Laura Perez-Beltrachini. A Serious Game for Second Language Acquisition in a Virtual Environment. *Journal on Systemics, Cybernetics and Informatics,*, 10(1):24–34, 2012.

[Anand and Kahn, 1992] Tej Anand and Gary Kahn. Making sense of gigabytes: a system for knowledge-based market analysis. In *Proceedings of the fourth conference on Innovative applications of artificial intelligence*, pages 57–69. AAAI Press, 1992.

[Androutsopoulos *et al.*, 2013] Ion Androutsopoulos, Gerasimos Lampouras, and Dimitrios Galanis. Generating natural language descriptions from owl ontologies: the naturalowl system. *Journal of Artificial Intelligence Research*, pages 671–715, 2013.

[Angeli *et al.*, 2010] Gabor Angeli, Percy Liang, and Dan Klein. A Simple Domain-independent Probabilistic Approach to Generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512. Association for Computational Linguistics, 2010.

[Baker *et al.*, 1998] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *COLING-ACL '98: Proceedings of the Conference*, pages 86–90, Montreal, Canada, 1998.

[Ballesteros *et al.*, 2015] Miguel Ballesteros, Bernd Bohnet, Simon Mille, and Leo Wanner. Data-driven sentence generation with non-isomorphic trees. In *Proceed-*

ings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT 2015), 2015.

[Banik *et al.*, 2012] Eva Banik, Claire Gardent, Donia Scott, Nikhil Dinesh, and Fennie Liang. KBGen: Text Generation from Knowledge bases as a New Shared Task. In *Proceedings of the seventh International Natural Language Generation Conference*, pages 141–145. Association for Computational Linguistics, 2012.

[Banik *et al.*, 2013] Eva Banik, Claire Gardent, and Eric Kow. The KBGen Challenge. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 94–97, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[Barker *et al.*, 2001] Ken Barker, Bruce Porter, and Peter Clark. A Library of Generic Concepts for Composing Knowledge Bases. In *Proceedings of the 1st international conference on Knowledge capture*, pages 14–21. ACM, 2001.

[Basile and Bos, 2011] Valerio Basile and Johan Bos. Towards generating text from discourse representation structures. In *Proceedings of the 13th European Workshop on Natural Language Generation*, ENLG '11, pages 145–150, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Belz *et al.*, 2011] Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France, 2011.

[Belz, 2007] Anja Belz. Probabilistic generation of weather forecast texts. Association for Computational Linguistics, 2007.

[Belz, 2008] Anja Belz. Automatic Generation of Weather Forecast Texts using Comprehensive Probabilistic Generation-Space Models. *Natural Language Engineering*, 14(4):431–455, 2008.

[Bohnet *et al.*, 2010] Bernd Bohnet, Leo Wanner, Simon Mille, and Alicia Burga. Broad coverage multilingual deep sentence generation with a stochastic multi-level realizer. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 98–106. Association for Computational Linguistics, 2010.

[Bontcheva and Wilks., 2004] K. Bontcheva and Y. Wilks. Automatic Report Generation from Ontologies: the MIAKT Approach. In *Ninth International Conference on Applications of Natural Language to Information Systems (NLDB'2004).* Lecture Notes in Computer Science 3136, Springer, Manchester, UK, 2004.

[Bontcheva, 2005] Kalina Bontcheva. Generating tailored textual summaries from ontologies. In *The Semantic Web: Research and Applications*, pages 531–545. Springer, 2005.

[Briscoe and Carroll, 1997] Ted Briscoe and John Carroll. Automatic extraction of subcategorization from corpora. In *Proceedings of the fifth conference on Applied natural language processing*, pages 356–363. Association for Computational Linguistics, 1997.

[Busemann, 1996] Stephan Busemann. Best-first surface realization. *arXiv preprint cmp-lg/9605010*, 1996.

[Butler *et al.*, 2013] Keith Butler, Priscilla Moraes, Ian Tabolt, and Kathy McCoy. Team udel kbgen 2013 challenge. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 206–207, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[Cahill and Van Genabith, 2006] Aoife Cahill and Josef Van Genabith. Robust pcfg-based generation using automatically acquired lfg approximations. In *Proceedings of the 21st International Conference on Computational Linguistics (COLING) and the 44th annual meeting of the Association for Computational Linguistics (ACL)*, pages 1033–1040, Sydney, Australia, 2006.

[Cahill *et al.*, 2007] Aoife Cahill, Martin Forst, and Christian Rohrer. Designing features for parse disambiguation and realisation ranking. 2007.

[Callaway, 2003] Charles B. Callaway. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 811–817, Acapulco, Mexico, 2003.

[Carenini *et al.*, 1994] Giuseppe Carenini, Vibhu O Mittal, and Johanna D Moore. Generating patient-specific interactive natural language explanations. In *Proceedings of the Eighteenth Annual Symposium on Computer Application in Medical Care*, page 5. American Medical Informatics Association, 1994.

[Carroll and Oepen, 2005] John Carroll and Stephan Oepen. High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP)*, Jeju Island, Korea, 2005.

[Chaudhri *et al.*, 2013] Vinay K Chaudhri, Michael A Wessel, and Stijn Heymans. KB_Bio_101: A Challenge for OWL Reasoners. In *ORE*, pages 114–120, 2013.

[Chen and Mooney, 2008] David L Chen and Raymond J Mooney. Learning to Sportscast: A Test of Grounded Language Acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM, 2008.

[Cheung and Penn, 2014] Jackie Chi Kit Cheung and Gerald Penn. Unsupervised sentence enhancement for automatic summarization. In *Proceedings of EMNLP*, pages 775–786, 2014.

[Chiang, 2000] David Chiang. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 456–463. Association for Computational Linguistics, 2000.

[Cimiano *et al.*, 2013] Philipp Cimiano, Janna Lüker, David Nagel, and Christina Unger. Exploiting ontology lexica for generating natural language texts from rdf data. 2013.

[Copestake and Flickinger, 2000] Ann Copestake and Dan Flickinger. An open source grammar development environment and broad-coverage english grammar using hpsg. In *IN PROCEEDINGS OF LREC 2000*, pages 591–600, 2000.

[Dahl *et al.*, 1994] Deborah A Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. Expanding the Scope of the ATIS Task: The ATIS-3 Corpus. In *Proceedings of the workshop on Human Language Technology*, pages 43–48. Association for Computational Linguistics, 1994.

[Danlos, 1989] Laurence Danlos. Some issues in text generation. In *Proceedings of Procesamiento del Lenguaje Natural*, pages 27–44. Sociedad Española para el Procesamiento del Lenguaje Natural, 1989.

[Dethlefs and Cuayáhuitl, 2012a] Nina Dethlefs and Heriberto Cuayáhuitl. Comparing hmms and bayesian networks for surface realisation. In *Proceedings of the 2012*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT '12, pages 636–640, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[Dethlefs and Cuayáhuitl, 2012b] Nina Dethlefs and Heriberto Cuayáhuitl. Comparing hmms and bayesian networks for surface realisation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 636–640. Association for Computational Linguistics, 2012.

[DeVault *et al.*, 2008a] D. DeVault, D. Traum, and Ron Artstein. Practical Grammar-Based NLG from Examples. In *The Fifth International Natural Language Generation Conference (INLG 2008)*, Ohio, June, 2008.

[DeVault *et al.*, 2008b] David DeVault, David Traum, and Ron Artstein. Making grammar-based generation easier to deploy in dialogue systems. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, pages 198–207. Association for Computational Linguistics, 2008.

[Doran *et al.*, 1994] Christy Doran, Dania Egedi, Beth Ann Hockey, Bangalore Srinivas, and Martin Zaidel. Xtag system: a wide coverage grammar for english. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pages 922–928. Association for Computational Linguistics, 1994.

[Duma and Klein, 2013] Daniel Duma and Ewan Klein. *Generating Natural Language from Linked Data: Unsupervised Template Extraction*, pages 83–94. ASSOC COMPUTATIONAL LINGUISTICS-ACL, 2013.

[Ehrig and Sure, 2004] Marc Ehrig and York Sure. Ontology mapping–an integrated approach. In *The Semantic Web: Research and Applications*, pages 76–91. Springer, 2004.

[Elhadad, 1993] Michael Elhadad. FUF: The universal unifier — User Manual, version 5.2. Technical report, Ben Gurion University of the Negev, 1993.

[Ell and Harth, 2014] Basil Ell and Andreas Harth. A language-independent method for the extraction of rdf verbalization templates. *INLG 2014*, page 26, 2014.

[Evans and Power, 2003] Roger Evans and Richard Power. Wysiwym: Building user interfaces with natural language feedback. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 203–206. Association for Computational Linguistics, 2003.

[Filippova and Strube, 2007] Katja Filippova and Michael Strube. Generating constituent order in german clauses. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, volume 45, 2007.

[Franconi *et al.*, 2010] E. Franconi, P. Guagliardo, and M. Trevisan. An Intelligent Query Interface Based on Ontology Navigation. In *Workshop on Visual Interfaces to the Social and Semantic Web, VISSW*. Citeseer, 2010.

[Franconi *et al.*, 2011] E. Franconi, P. Guagliardo, S. Tessaris, and M. Trevisan. Quelo: An Ontology-Driven Query Interface. In *Proceedings of the 24th International Workshop on Description Logics (DL 2011)*, 2011.

[Frank, 2002] Robert Frank. Phrase structure composition and syntactic dependencies. 2002.

[Galanis *et al.*, 2009] D. Galanis, G. Karakatsiotis, G. Lampouras, and I. Androutsopoulos. An Open-Source Natural Language Generator for OWL Ontologies and its use in Protégé and Second Life. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, pages 17–20. Association for Computational Linguistics, 2009.

[Gali and Venkatapathy, 2009] Karthik Gali and Sriram Venkatapathy. Sentence realisation from bag of words with dependency constraints. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 19–24. Association for Computational Linguistics, 2009.

[Gardent and Kallmeyer, 2003] Claire Gardent and Laura Kallmeyer. Semantic construction in feature-based tag. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 123–130. Association for Computational Linguistics, 2003.

[Gardent and Kow, 2005] Claire Gardent and Eric Kow. Generating and selecting grammatical paraphrases. In *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG)*, Aberdeen, Scotland, Aug 2005.

[Gardent *et al.*, 2007] Claire Gardent, Eric Kow, et al. A symbolic approach to near-deterministic surface realisation using tree adjoining grammar. In *ACL*, volume 7, pages 328–335, 2007.

[Gerdemann and Hinrichs, 1990] Dale Gerdemann and Erhard W Hinrichs. Functor-driven natural language generation with categorial-unification grammars. In *Proceedings of the 13th conference on Computational linguistics-Volume 2*, pages 145–150. Association for Computational Linguistics, 1990.

[Goldberg *et al.*, 1994] Eli Goldberg, Norbert Driedger, Richard Kittredge, et al. Using Natural-Language Processing to Produce Weather Forecasts. *IEEE Expert*, 9(2):45–53, 1994.

[Gruber, 1993] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993.

[Guarino *et al.*, 2009] Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an ontology? In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 1–17. Springer Berlin Heidelberg, 2009.

[Gunning *et al.*, 2010] David Gunning, Vinay K Chaudhri, Peter E Clark, Ken Barker, Shaw-Yi Chaw, Mark Greaves, Benjamin Grosof, Alice Leung, David D McDonald, Sunil Mishra, et al. Project Halo Update—Progress Toward Digital Aristotle. *AI Magazine*, 31(3):33–58, 2010.

[Guo *et al.*, 2008] Yuqing Guo, Josef van Genabith, and Haifeng Wang. Dependency-based n-gram models for general purpose sentence realisation. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 297–304, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Guo *et al.*, 2011] Yuqing Guo, Deirdre Hogan, and Josef van Genabith. Dcu at generation challenges 2011 surface realisation track. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 227–229, Nancy, France, September 2011. Association for Computational Linguistics.

[Gyawali and Gardent, 2013] Bikash Gyawali and Claire Gardent. LOR-KBGEN, A Hybrid Approach to Generating from the KBGen Knowledge-Base. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 204–205, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[Gyawali, 2011] Bikash Gyawali. Answering Factoid Questions via Ontologies : A Natural Language Generation Approach. Master's thesis, University of Malta and Université de Lorraine, 2011.

[Hajič *et al.*, 2009] Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria A. Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics, 2009.

[Hallett *et al.*, 2007] Catalina Hallett, Donia Scott, and Richard Power. Composing questions through conceptual authoring. *Computational Linguistics*, 33(1):105–133, 2007.

[Hinton, 2002] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[Hockenmaier and Steedman, 2007] Julia Hockenmaier and Mark Steedman. Ccgbank: A corpus of ccg derivations and dependency structures extracted from the penn treebank. *Comput. Linguist.*, 33(3):355–396, September 2007.

[Kaljurand and Fuchs, 2007] K. Kaljurand and N.E. Fuchs. Verbalizing OWL in Attempto Controlled English. *Proceedings of OWLED07*, 2007.

[Kan and McKeown, 2002] Min-Yen Kan and Kathleen R. McKeown. Corpus-trained text generation for summarization, 2002.

[Kay, 1996] Martin Kay. Chart generation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 200–204. Association for Computational Linguistics, 1996.

[Kim and Mooney, 2010] Joohyun Kim and Raymond J Mooney. Generative alignment and semantic parsing for learning from ambiguous supervision. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 543–551. Association for Computational Linguistics, 2010.

[Kim *et al.*, 2003a] J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. Genia Corpus—A Semantically Annotated Corpus for Bio-Textmining. *Bioinformatics*, 19(suppl 1):i180–i182, 2003.

[Kim *et al.*, 2003b] J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182, 2003.

[King *et al.*, 2003] Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. The parc 700 dependency bank. In *In Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03*, pages 1–8, 2003.

[Kipper *et al.*, 2000] Karin Kipper, Hoa Trang Dang, and Martha Palmer. Class-based construction of a verb lexicon. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 691–696. AAAI Press, 2000.

[Kondadadi *et al.*, 2013] Ravi Kondadadi, Blake Howald, and Frank Schilder. A statistical nlg framework for aggregated planning and realization. 2013.

[Konstas and Lapata, 2012a] Ioannis Konstas and Mirella Lapata. Concept-to-Text Generation via Discriminative Reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 369–378. Association for Computational Linguistics, 2012.

[Konstas and Lapata, 2012b] Ioannis Konstas and Mirella Lapata. Unsupervised Concept-to-Text Generation with Hypergraphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 752–761. Association for Computational Linguistics, 2012.

[Korhonen, 2002] Anna Korhonen. Semantically motivated subcategorization acquisition. In *Proceedings of the ACL-02 Workshop on Unsupervised Lexical Acquisition - Volume 9*, ULA '02, pages 51–58, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[Kow and Belz, 2012] Eric Kow and Anja Belz. Lg-eval: A toolkit for creating online language evaluation experiments. In *LREC*, pages 4033–4037, 2012.

[Lafferty, 2001] John Lafferty. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282–289. Morgan Kaufmann, 2001.

[Liang *et al.*, 2009] Percy Liang, Michael I Jordan, and Dan Klein. Learning Semantic Correspondences with Less Supervision. In *Proceedings of the Joint Conference* *of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 91–99. Association for Computational Linguistics, 2009.

[Lopez *et al.*, 2007] Vanessa Lopez, Victoria Uren, Enrico Motta, and Michele Pasin. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):72–105, 2007.

[Lu and Ng, 2011] Wei Lu and Hwee Tou Ng. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1611–1622. Association for Computational Linguistics, 2011.

[Lu *et al.*, 2009] Wei Lu, Hwee Tou Ng, and Wee Sun Lee. Natural language generation with tree conditional random fields. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 400–409. Association for Computational Linguistics, 2009.

[Marciniak and Strube, 2005] Tomasz Marciniak and Michael Strube. Using an annotated corpus as a knowledge source for language generation. In *Proceedings of the Corpus Linguistics'05 Workshop Using Corpora for NLG (UNNLG-2005*, 2005.

[McRoy *et al.*, 2003] Susan W. McRoy, Songsak Channnarukul, and Syed S. Ali. An augmented template-based approach to text realization. *Natural Language Engineering*, 9:381–420, 12 2003.

[Mellish and Pan, 2006] Chris Mellish and Jeff Z Pan. Finding subsumers for natural language presentation. In *2006 International Workshop on Description Logics DL'06*, page 127, 2006.

[Mihăilă *et al.*, 2013] C. Mihăilă, T. Ohta, S. Pyysalo, and S. Ananiadou. Biocause: Annotating and Analysing Causality in the Biomedical Domain. *BMC BioInformatics*, 2013.

[Nakanishi *et al.*, 2005] Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. Probabilistic models for disambiguation of an hpsg-based chart generator. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 93–102. Association for Computational Linguistics, 2005.

[Narayan and Gardent, 2012a] S. Narayan and C. Gardent. Structure-Driven Lexicalist Generation. In *Proceedings of the 24th International Conference in Computational Linguistics*, pages 100–113, Mumbai, Inde, 2012.

[Narayan and Gardent, 2012b] Shashi Narayan and Claire Gardent. Structure-driven lexicalist generation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 2027–2042, Mumbai, India, December 2012. The COLING 2012 Organizing Committee.

[Papasalouros *et al.*, 2008] Andreas Papasalouros, Konstantinos Kanaris, and Konstantinos Kotis. Automatic Generation of Multiple Choice Questions from Domain Ontologies. Amsterdam, 2008.

[Papineni *et al.*, 2002] K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[Paris, 1988] C.L. Paris. Tailoring Object Descriptions to a User's Level of Expertise. *Computational Linguistics*, 14(3):64–78, 1988.

[Perez-Beltrachini *et al.*, 2014] Laura Perez-Beltrachini, Claire Gardent, and Enrico Franconi. Incremental query generation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Gothenburg, Sweden, April 2014.

[Power and Third, 2010] R. Power and A. Third. Expressing OWL Axioms by English Sentences: Dubious in Theory, Feasible in Practice. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1006–1013. Association for Computational Linguistics, 2010.

[Pyysalo *et al.*, 2007] Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. Bioinfer: A Corpus for Information Extraction in the Biomedical Domain. *BMC Bioinformatics*, 2007.

[Rajkumar and White, 2014] Rajakrishnan Rajkumar and Michael White. Better surface realization through psycholinguistics. *Language and Linguistics Compass*, 8(10):428–448, 2014.

[Rajkumar *et al.*, 2011] Rajakrishnan Rajkumar, Dominic Espinosa, and Michael White. The osu system for surface realization at generation challenges 2011.

In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, pages 236–238, Nancy, France, 2011.

[Ratnaparkhi, 2000] Adwait Ratnaparkhi. Trainable methods for surface natural language generation. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, pages 194–201, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[Regneri *et al.*, 2013] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics (TACL)*, 1:25–36, 2013.

[Reiter and Dale, 2000] Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Studies in Natural Language Processing. Cambridge University Press, 2000.

[Reiter *et al.*, 2003] E. Reiter, R. Robertson, and L.M. Osman. Lessons from a Failure: Generating Tailored Smoking Cessation Letters. *Artificial Intelligence*, 144(1):41–58, 2003.

[Reiter *et al.*, 2005] Ehud Reiter, Somayajulu Sripada, Jim Hunter, and Ian Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169, 2005.

[Rimell *et al.*, 2013] Laura Rimell, Thomas Lippincott, Karin Verspoor, Helen L Johnson, and Anna Korhonen. Acquisition and evaluation of verb subcategorization resources for biomedicine. *Journal of biomedical informatics*, 46(2):228–237, 2013.

[Ringger *et al.*, 2004] Eric Ringger, Michael Gamon, Robert C Moore, David Rojas, Martine Smets, and Simon Corston-Oliver. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th international conference on Computational Linguistics*, page 673. Association for Computational Linguistics, 2004.

[Russell *et al.*, 1990] Graham Russell, Susan Warwick, and John Carroll. Asymmetry in parsing and generating with unification grammars: Case studies from elu. In *Proceedings of the 28th Annual Meeting on Association for Computational Linguistics*, ACL '90, pages 205–211, Stroudsburg, PA, USA, 1990. Association for Computational Linguistics.

[Sarkar and Zeman, 2000] Anoop Sarkar and Daniel Zeman. Automatic extraction of subcategorization frames for czech. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 691–697. Association for Computational Linguistics, 2000.

[Shemtov, 1996] Hadar Shemtov. Generation of paraphrases from ambiguous logical forms. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 919–924. Association for Computational Linguistics, 1996.

[Shieber *et al.*, 1990] Stuart M Shieber, Gertjan Van Noord, Fernando CN Pereira, and Robert C Moore. Semantic-head-driven generation. *Computational Linguistics*, 16(1):30–42, 1990.

[Stevens *et al.*, 2011] R. Stevens, J. Malone, S. Williams, R. Power, and A. Third. Automating Generation of Textual Class Definitions from OWL to English. *Journal of Biomedical Semantics*, 2(Suppl 2):S5, 2011.

[Stolcke, 2002] Andreas Stolcke. SRILM-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 257–286, November 2002.

[Sun and Mellish, 2006] Xiantang Sun and Chris Mellish. Domain independent sentence generation from rdf representations for the semantic web. In *Combined Workshop on Language-Enabled Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems*. European Conference on AI, 2006.

[Swier and Stevenson, 2004] Robert S Swier and Suzanne Stevenson. Unsupervised Semantic Role Labelling. In *Proceedings of EMNLP*, volume 95, page 102, 2004.

[Tennant *et al.*, 1989] H.R. Tennant, K.M. Ross, and R.M. Saenz. Menu-Based Natural Language Understanding System, May 9 1989. US Patent 4,829,423.

[Thompson *et al.*, 2009] P. Thompson, S. A. Iqbal, J. McNaught, and S. Ananiadou. Construction of an Annotated Corpus to Support Biomedical Information Extraction. *BMC Bioinformatics*, 2009.

[Toutanova and Manning, 2002] Kristina Toutanova and Christopher D Manning. Feature selection for a rich hpsg grammar using decision trees. In *proceedings of the 6th conference on Natural language learning-Volume 20*, pages 1–7. Association for Computational Linguistics, 2002.

[Trevisan, 2010] M. Trevisan. *A Portable Menuguided Natural Language Interface to Knowledge Bases for Querytool.* PhD thesis, Master thesis, Free University of Bozen-Bolzano (Italy) and University of Groningen (Netherlands), 2010.

[Van Deemter and Odijk, 1997] Kees Van Deemter and Jan Odijk. Context modeling and the generation of spoken discourse. *Speech Communication*, 21(1):101–121, 1997.

[Van Deemter *et al.*, 2005] Kees Van Deemter, Emiel Krahmer, and Mariët Theune. Real versus template-based natural language generation: A false opposition? *Computational Linguistics*, 31(1):15–24, 2005.

[Varges and Mellish, 2001] Sebastian Varges and Chris Mellish. Instance-based natural language generation. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics, 2001.

[Walter *et al.*, 2013] Sebastian Walter, Christina Unger, and Philipp Cimiano. A corpus-based approach for the induction of ontology lexica. In *Natural Language Processing and Information Systems*, pages 102–113. Springer, 2013.

[Wang and Zhang, 2012] Rui Wang and Yi Zhang. Sentence realization with unlexicalized tree linearization grammars. In *24th International Conference on Computational Linguistics*, page 1301, 2012.

[Wang *et al.*, 2015] Y. Wang, J. Berant, and P. Liang. Building a semantic parser overnight. In *Association for Computational Linguistics (ACL)*, 2015.

[Wang, 1980] Juen-tin Wang. On computational sentence generation from logical form. In *Proceedings of the 8th conference on Computational linguistics*, pages 405–411. Association for Computational Linguistics, 1980.

[White and Rajkumar, 2009] Michael White and Rajakrishnan Rajkumar. Perceptron reranking for ccg realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 410–419. Association for Computational Linguistics, 2009.

[White *et al.*, 2007] Michael White, Rajakrishnan Rajkumar, and Scott Martin. Towards broad coverage surface realization with ccg. In *In Proc. of the Workshop on Using Corpora for NLG: Language Generation and Machine Translation (UC-NLG+MT*, 2007.

[Wilcock, 2003] G. Wilcock. Talking OWLs: Towards an Ontology Verbalizer. *Human Language Technology for the Semantic Web and Web Services, ISWC*, 3:109–112, 2003.

[Williams and Power, 2010] Sandra Williams and Richard Power. Grouping Axioms for More Coherent Ontology Descriptions. In *Proceedings of the 6th International Natural Language Generation Conference (INLG 2010)*, pages 197–202, Dublin, 2010.

[Wong and Mooney, 2006] Yuk Wah Wong and Raymond J. Mooney. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 439–446, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[Wong and Mooney, 2007] Yuk Wah Wong and Raymond J Mooney. Generation by Inverting a Semantic Parser that Uses Statistical Machine Translation. In *HLT-NAACL*, pages 172–179, 2007.

[Zarrieß *et al.*, 2011] Sina Zarrieß, Aoife Cahill, and Jonas Kuhn. Underspecifying and predicting voice for surface realisation ranking. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1007–1017. Association for Computational Linguistics, 2011.

[Zarrieβ and Richardson, 2013] Sina Zarrieβ and Kyle Richardson. An automatic method for building a data-to-text generator. In *Proceedings of the 14th European Workshop on Natural Language Generation*, Sofia, Bulgaria, August 2013.

[Zhong and Stent, 2009] Huayan Zhong and Amanda Stent. Determining the position of adverbial phrases in english. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 229–232. Association for Computational Linguistics, 2009.