Lexical choice in Abstract Dependency Trees

Dieke Oele
Rijksuniversiteit Groningen,
Groningen,
d.oele@rug.nl

Gertjan van Noord
Rijksuniversiteit Groningen,
Groningen,
g.j.m.van.noord@rug.nl

Abstract

In this work lexical choice in generation for Machine Translation is explored using lexical semantics. We address this problem by replacing lemmas with synonyms in the abstract representations that are used as input for generation, given a WordNet synset. In order to find the correct lemma for each node we propose to map dependency trees to Hidden Markov Trees that describe the probability of a node given its parent node. A tree-modified Viterbi algorithm is then utilized to find the most probable hidden tree containing the correct lemmas given their context. The model is implemented in a Machine Translation system for English to Dutch. The output sentences, generated from the modified dependency structures, contained a lot of erroneous substituted words. This is mainly due to the fact that a large amount of synsets, used as input for the model, are incorrect. The input to the model now contains the synset that is most frequent given the lemma in general, not the optimal synset given the domain of the sentences. We therefore propose to implement a domain specific WSD-system in our pipeline in future work.

1 Introduction

This paper addresses the problem of lexical choice in the generation phase of a deep Machine Translation (MT) system using a Hidden Markov Tree Model (HMTM). In Natural Language Generation (NLG), an abstract representation is transformed into one or more linguistic utterances. Lexical choice is a subtask of this process, in which lemmas need to be chosen to adequately express the content of the intended utterance.

For the generation component of a deep MT-system, the challenge lies in the construction of sentences on the basis of deep representations. The process in this setup includes two major steps: the construction of deep structures for the sentence to be generated and the realization of the sentence on the basis of the grammar which will ensure that the created structure conforms to all the requirements for a complete structure with respect to the grammar formalism. The second step entails the selection of appropriate lexical units and the application of syntactic rules.

In a deep transfer-based MT system the problem of lexical choice also needs to be addressed. The choice of a correct lemma is a very difficult task that depends heavily on the quality of the dictionaries used. WordNet (Fellbaum, 1998) could be seen as such a dictionary, where each synset has its own definition. A WordNet is a lexical semantic database containing lemmas corresponding to their word meanings including the most general and central part of the language. Querying WordNet for a word returns a group of one or more synonyms called a synset. Those synsets contain a set of words of the same class, which are roughly synonymous in one of their

This work is licenced under a Creative Commons Attribution 4.0 International License. Page numbers and proceedings footer are added by the organizers. License details: http://creativecommons.org/licenses/by/4.0/

Proceedings of the 1st Deep Machine Translation Workshop (DMTW 2015), pages 73–80,
Praha, Czech Republic, 3–4 September 2015.

meanings. They are well suited for lexical choice, as they contain sets of lemmas that are synonymous in specific contexts. Unfortunately, not every lemma in a synset is a full synonym of its original word which could cause severe errors when selecting the most probable variant for each node without considering the context. We therefore claim that lexical choice should take into account both context and distributional information.

Consider for instance the English word "free" in example 1. In the current MT-system for English to Dutch, this word is always translated with "vrij". This is a satisfactory decision in example 1 where the meaning of "not occupied" is required. In the sentence in 2, however, "free" means "free of charge" which, in Dutch, should be translated with "gratis". It would therefore be useful if the input to the system would contain the meaning of a target lemma instead of a literal translation of the source lemma, for example in the form of a WordNet synset. This information could be used to find an appropriate lemma given its meaning.

(1) You can leave some **free** space in your shared folders.
→ *Je kan wat* **vrije** *ruimte overlaten in je gedeelde mappen.*

(2) This is **free** antivirus software.
→ *Dit is* ***vrije** *antivirus software.*

A similar error occurs in example 3 where the word "limiet" is translated with "grens". This translation fits in this particular context, both meaning "border", in example 4, however, the meaning of a quantitative limit is required. In both cases the chosen word by the system is the most frequent option and is therefore selected in both contexts.

(3) There have to be **limits** to all things.
→ *Overal zijn* **grenzen** *aan.*

(4) Having a large number of of shared folders can occupy your space **limit**.
→ *Als je veel gedeelde mappen hebt kan dat je ruimte* ***grens** *beperken.*

.

These examples indicate that a correct translation requires a lexical choice process that can determine which translation of a source word is most appropriate in its target context. In transfer-based MT the task of lexical choice can be formulated as follows: given a semantic or conceptual specification, find its best realization. We can define the process of lexical choice as the operations of deleting, modifying or adding lexical units in order to form more natural sentences with a correct meaning.

The problem of lexical choice in MT has not yet been investigated thoroughly, probably due to the fact that its output is hard to evaluate. For example, when a different lemma is returned than the one from the gold standard it might still be appropriate to the context but marked as an error by the evaluation method. Stede (1993) was the first to recognize the need to involve semantic context. A number of algorithms and models have been developed for lexical choice, for example Edmonds and Hirst (2002) developed a model for choosing between words with similar core meanings but with different connotations.

WordNet has not often been used as a dictionary for lexical choice in generation, even though work exists on the usefulness of such a resource for NLG-related tasks such as domain adaptation and paraphrasing (Jing, 1998). For instance, Basile (2014) proposed an unsupervised algorithm for lexical choice from WordNet synsets called Ksel that exploits the WordNet hierarchy of hypernyms/hyponyms to produce the most appropriate lemma for a given synset.

Also, the use of Hidden Markov Tree models for lexical choice in Wordnet synsets is novel. Crouse et al. (1996) introduced the adaptation of Hidden Markov Chains to tree models for

signal processing. The corresponding adaptation of the classic Viterbi algorithm, used to restore the hidden state tree, was introduced by Durand et al. (2004). Previous applications of the tree model are: image segmentation, signal classification, denoising and image document categorization (Durand et al., 2004). The use of these models in natural language processing is fairly new and has been applied to word alignment (Kondo et al., 2013) and Machine Translation (Žabokrtský and Popel, 2009). The latter was the first to apply HMTMs to lexical choice using a variant of the Viterbi algorithm in the transfer phase of a deep-syntax based machine translation system.

To tackle the problem of lexical choice we propose the mapping of a dependency structure over synsets to a dependency structure over lemmas while taking into account both context information as the frequency of the lemma and synset combination. A dependency tree is a labeled tree in which nodes correspond to the words of a sentence. It contains edges that represent the grammatical relations between those words, such as nominal subject, direct object or determiner. We map the lemma of each content word in the tree to a WordNet synset and subsequently find a correct substitution based on a target language synset. The goal is to improve the output of a Machine Translation (MT) system built on deeper semantic engineering approaches.

The independence assumptions made by Markov Tree Models can be useful for modeling syntactic trees. They fit dependency trees well, since they assume conditional dependence (in the probabilistic sense) only along tree edges, which corresponds to intuition behind dependency relations (in the linguistic sense) in dependency trees. Moreover, analogously to applications of HMMs on sequence labeling, HMTMs can be used for labeling nodes of a dependency tree, interpreted as revealing the hidden states in the tree nodes, given another (observable) labeling of the nodes of the same tree by use of a tree-modified Viterbi algorithm.

This paper is structured as follows. In Section 2, we introduce the HMTM model for lexical choice and the modified Tree-Viterbi algorithm. Section 3 gives a brief description of experiments that test the model. Then, section 4 discusses the obtained results and possible improvements. Ultimately, this paper is concluded in Section 5.

2 Method

The first part of this section contains a brief description of Hidden Markov Tree Models. Then, the tree-viterbi algorithm for lexical choice is introduced.

2.1 Hidden Markov Tree Models

HMTMs are analogous to well known Hidden Markov Models (HMM). However, instead of a linear chain of observations and their corresponding hidden states they map over a tree of observations. They are similar to Hidden Markov chains given the fact that they both contain a sequence of observed states with corresponding hidden states. Furthermore, they both rely on transition probabilities and emission probabilities. Like HMMs, HMTMs are used with two main algorithms. Namely a smoothing algorithm that calculates the probabilities of being in state j at node n given the observed data, and a global restoration algorithm. More information about HMTMs can be found in Diligenti et al. (2003) and in Durand et al. (2004).

In the Markov process we use for the choice of lemmas, we assume that we are given a directed dependency tree. The tree is defined by an observed tree containing synsets in their nodes, $W = \{W(n_1), ..., W(n_m)\}$, and a hidden tree containing target lemmas, $T = \{T(n_1), ..., T(n_m)\}$, isomorphic to the observed tree where m is the size of the tree. The function $\pi : 1, ..., N \rightarrow 0, ..., N$, $\pi(n)$ represents the unique parent of node n with 0 corresponding to the root of the tree. Each node, except the root node, refers to a word of in the sentence. Like HMMs, HMTMs make two independence assumptions: given $T(\pi(n))$, $T(n)$ is condi-

tionally independent of other nodes and given $T(n)$, $W(n)$ is conditionally independent of other nodes. From these assumptions, we obtain the following distribution on pairs (W, T) of observed synsets and target lemmas:

$$p(w,t) = \sum_{n=1}^{N} p\left(t_n \mid t_{\pi(n)}\right) p\left(w_n \mid t_n\right) \tag{1}$$

When using HMTM for lexical choice, the hidden states consist of actual lemmas, whereas the observations are word senses (synsets). Analogously to regular Hidden Markov Models, HTMTs are defined by the following parameters:

Transition probabilities:

- $P(hiddenstate|hiddenstate)$

Emission probabilities:

- $P(observation|hiddenstate)$

The transition probabilities of a lemma w given a grammatical relation rel and its parent p can be collected from large parsed corpora. For example, if we want the probability of the lemma "beer" given a parent "drink" in the dependency relation "obj":

$$p\left(beer \mid obj1, drink\right) = \frac{freq(drink, obj1, beer)}{freq(drink, obj)} \tag{2}$$

The frequency of a lemma given its parent is the count of how often its parent appears in relation rel and N is the total number of p as arguments of rel. For example, if "drink" occurs 40 times with an object, and in 20 cases that object is the lemma "beer", then we estimate the probability as 0.5.

The emission probabilities can be estimated from sense annotated corpora. We need to estimate the probability of an observed output (the sense), given the hidden state (the lemma). Consider for example the chance of the hidden state "beer" given its synset:

$$P(\{lager, beer, ale, ...\}|beer) \tag{3}$$

If the lemma "ale" is associated in the corpus with the "beer" sense in 89 out of a 100 cases, then the emission probability will be estimated as 0.89.

2.2 Tree-Viterbi

Durand et al. (2004) claim that it is not possible to involve a downward recursion starting at the root state of the tree due to the fact that this would require the results of the upward recursion. The main difference between a tree-viterbi as opposed to its original one is therefore that it starts at its leaf nodes and continues upwards. In every node of each state and each of its children, a downward pointer to the optimal hidden state of the child is stored. Downward recursion is then used along the pointers from the optimal root state in order to retrieve the most probable hidden tree.

3 Experiments

In this section, we present the data and the systems that were used for the experiments. In the experiments the tree-modified Viterbi algorithm for lexical choice is applied to the dependency structures that are used for generation in a deep MT-system. Subsequently the output, containing dependency trees, is used as input to the generator.

3.1 Data

The system for lexical choice is implemented in a machine translation system and tested on Batch 1 of the QTLeap Corpus Osenova et al. (2015). This IT help desk scenario contains translations of customer data from Portuguese into each of the project languages.

3.2 Systems

The sentences are analyzed and translated from English to Dutch with Treex, a modular framework for natural language processing (Popel and Žabokrtský, 2010). It contains a tree-to-tree machine translation system whose translation process follows the analysis-transfer-synthesis pipeline (Žabokrtský and Popel, 2009).

In the analysis phase, a source sentence is transformed into a deep syntax dependency representation which is mapped to the target language. Isomorphism of the tree representation is mostly assumed in both languages, translating the tree node-by-node. In the English to Dutch pipeline, the resulting dependency trees are transferred to Dutch abstract abstract representations that are the input for the generation of Dutch sentences.

The Alpino system for Dutch (van Noord, 2006) is a collection of tools and programs for parsing Dutch sentences into dependency structures, and for generating Dutch sentences on the basis of an abstraction of dependency structures. Since dependency structures for generation contain less information (such as word order and word inflection) than dependency trees, we refer to them as Abstract Dependency Trees (ADT's) (De Kok, 2013). ADTs model the grammatical relations between lexical items and categories built from lexical items. Similar to a normal dependency tree, it contains a syntactical representation of a sentence in the form of a tree. In the Alpino Generator (De Kok, 2013), the grammar is used in the reverse direction as for parsing. The process starts with an abstract dependency structure and then uses the grammar to construct one or more sentences.

3.3 Setup

Before passing the abstract dependency trees through the tree-viterbi algorithm, first the linguistic parent is found for each node. Then, every lemma is matched with a synset. The model takes abstract dependency structures over senses as input. The current system, however, does not provide synsets for a given node. Therefore, a first step is necessary that maps abstract dependency structures over lemmas to abstract dependency over senses.

Synonyms of frequent senses of a source lemma are more likely to provide correct substitutions than synonyms of the lemmas infrequent senses. Therefore, in order to find the input synsets for each node, the most frequent synset given a lemma in a sense tagged corpus, in this case DutchSemCor (Vossen et al., 2012), is taken. These synsets now represent the observed state of the nodes. For these synsets the probability of their hidden states is computed. For example, the most frequent synset for the English lemma "dust" in sentence 5 appears with the following lemmas in the sense tagged corpus: $substantie, materie, stof$. These lemmas can be seen as the hidden states in the model. In this context the best choice would be "stof", and is up to the The tree-viterbi algorithm to choose this option over the other lemmas, given the synset and its context.

(5) **Dust** makes the computer cooling more difficult.

To find the different variants of the lemmas for replacement, the Dutch WordNet, Cornetto (Vossen et al., 2013), is used. A transition probability matrix is created from large parsed corpora that can be queried for each lemma given its parent lemma and their relation. The tree-viterbi algorithm is then applied on the trees to find the most probable lemmas given their

context and the frequency of the synsets. The lemmas in the optimal hidden tree are used to substitute the original lemma in the node. Ultimately, the trees are generated with the Alpino generator.

4 Results

From a manual evaluation of the results, it becomes clear that the system does not substitute a lot of lemmas. This is due to the fact that we only have counts for a limited amount of synsets in the sense annotated corpus. Some substitutions that are made can be considered satisfactory. For instance, the replacement of the adjective "simpel" (simple) to "eenvoudig" (simple) in sentence 6 maintains the meaning of the original sentence. However, the output mostly contains a lot of substitutions that are not considered correct, possibly changing the intended meaning of the target sentence. This happens for example when the system substitutes the noun "toets" (key) for "proef" (test) in sentence 7. Since the lemma "toets" is very common in the 1000 interactions of the corpus, it is substituted incorrectly 16 times. Other frequent words that are replaced with a wrong synonym are, for example, "controleren" (meaning "to check" in most of the contexts) is replaced with "beheersen" (to rule) 98 times while "raam" substitutes "venster" (window frame) 19 times.

(6) Je kan een **simpel [=> eenvoudig]** programma gebruiken.
You can use a **simple** *program.*

(7) Klik op de CTRL **toets [=> proef]**.
Click the CTRL **key [=> test]**.

One reason that a target synonym cannot substitute a source synset in some context is if the input synset appears in a different sense than the one in which it is synonymous with the target. In most cases, the algorithm chooses the same lemma. However, when the lemmas in a synset belong to the wrong sense, the system has a high chance of selection a wrong lemma.

Consider for example the lemma "menu", that appears in two Dutch synsets:

(8) a. {menukaart:noun:1', 'menu:noun:1', 'spijskaart:noun:1', 'kaart:noun:4'}
b. **{'menu:noun:3', 'keuzemenu:noun:1'}**

Since the data that was used for the experiment belongs to the IT domain, the second synset, in bold, is the preferred one. However, the first synset, with the meaning of *restaurant menu* is more frequent in the sense tagged corpus, as it was created in a general domain corpus, giving the option of substituting "menu" with "kaart" (map).

An interesting observation is that, if the target lemmas from these wrong senses are compared with other lemmas in the Viterbi-algorithm, they can cause each others to be replaced erroneously as well. This problem becomes apparent when looking at example 9.

(9) Klik op het pictogram waarop **"achtergrond beeld" [→ "toneel voorstelling"]** staat.
Click on the icon that says **"background image" [→ "theater performance"]**.

In this sentence, both lemmas "achtergrond" (background) and "beeld" (performance) have a bigger emission probability to be replaced by their original lemma. However, the combination of "theater" (which already is a doubtful synonym for "achtergrond" in any context) and "voorstelling", has a very frequent transition probability, causing an inaccurate substitution for both lemmas in this sentence.

Domain clearly is a problem when choosing the right sense. The frequency distribution of the senses of lemmas depends on the genre and domain of the text under consideration. A possible solution to finding the right synset, without using context information, is to use Word

Sense Disambiguation (WSD) from untagged text. This method aims to obtain, for each target word, the sense which is predominant in the target, domain-specific, corpus. McCarthy et al. (2004), for example, used such a corpus to construct a distributional thesaurus of related words. Subsequently, they disambiguated each target word using pairwise similarity measures based on WordNet, taking as pairs the target word and each of the most related words according to the distributional thesaurus up to a certain threshold. This method would not only allow our system to consider more lemmas for replacement, because more frequency information on synsets would then be available, it would also have a bigger chance of starting from correct input synsets. In future work we therefore intend to integrate this method in our pipeline.

Another problem that is highly likely to cause errors in the tree-viterbi algorithm are mistakes in the analysis and/or the transfer phase. For example, errors in the assignment of part-of-speech tags or dependency relations could have negative effects on the outcome since it would not be possible to find correct transition probabilities in the transition matrix. These errors should be solved in the analysis phase of the MT-pipeline and are therefore beyond the scope of this work.

5 Conclusion

In this work we intended to tackle the problem of lexical choice in order to improve the output of a Machine Translation system. To solve this we proposed the use of HMTMs for lexical choice. A dependency structure over synsets is mapped to a dependency structure over lemmas while taking into account both information of the context and the frequency of the lemma and synset combination. Although the obtained results contain some satisfactory substitutions, the system makes a lot of unwanted ones as well. These wrong substitutions are mostly due to the choice of the incorrect (most frequent) sense for a lemma in this particular domain. We therefore proposed the use of a method that first finds the right synset for a given lemma before applying the tree-viterbi algorithm.

Acknowledgements

This work has been supported by the European Union's Seventh Framework Programme for research, technological development and demonstration (QTLeap, grant agreement no 610516).

References

Basile, V. (2014). A lesk-inspired unsupervised algorithm for lexical choice from wordnet synsets. *The First Italian Conference on Computational Linguistics CLiC-it 2014*, page 48.

Crouse, M. S., Baraniuk, R. G., and Nowak, R. D. (1996). Hidden markov models for wavelet-based signal processing. In *Signals, Systems and Computers, 1996. Conference Record of the Thirtieth Asilomar Conference on*, pages 1029–1035. IEEE.

De Kok, D. (2013). *Reversible Stochastic Attribute-value Grammars*. Groningen dissertations in linguistics.

Diligenti, M., Frasconi, P., and Gori, M. (2003). Hidden tree markov models for document image classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(4):519–523.

Durand, J.-B., Goncalves, P., and Guédon, Y. (2004). Computational methods for hidden markov tree models-an application to wavelet trees. *Signal Processing, IEEE Transactions on*, 52(9):2551–2560.

Edmonds, P. and Hirst, G. (2002). Near-synonymy and lexical choice. *Comput. Linguist.*, 28(2):105–144.

Fellbaum, C., editor (1998). *WordNet An Electronic Lexical Database*. The MIT Press, Cambridge, MA ; London.

Jing, H. (1998). Applying wordnet to natural language generation. In *University of Montreal*. Citeseer.

Kondo, S., Duh, K., and Matsumoto, Y. (2013). Hidden markov tree model for word alignment. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, Sofia, Bulgaria. Association for Computational Linguistics.

McCarthy, D., Koeling, R., Weeds, J., and Carroll, J. (2004). Finding predominant word senses in untagged text. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Osenova, P., Gaudio, R. D., Silva, J., Burchardt, A., Popel, M., van Noord, G., Oele, D., and Labaka, G. (2015). Interim report on the curation of language resources and tools for deep mt. Technical Report Deliverable D2.5, Version 2.0, QTLeap Project.

Popel, M. and Žabokrtský, Z. (2010). Tectomt: Modular nlp framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing*, IceTAL'10, pages 293–304, Berlin, Heidelberg. Springer-Verlag.

Stede, M. (1993). Lexical choice criteria in language generation. In *Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics*, pages 454–459. Association for Computational Linguistics.

van Noord, G. (2006). **A**t **L**ast **P**arsing **I**s **N**ow **O**perational. In *TALN 2006 Verbum Ex Machina, Actes De La 13e Conference sur Le Traitement Automatique des Langues naturelles*, pages 20–42, Leuven.

Vossen, P., Görög, A., Izquierdo, R., and den Bosch, A. V. (2012). Dutchsemcor: Targeting the ideal sense-tagged corpus. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Vossen, P., van der Vliet, H., Maks, I., Segers, R., Moens, M.-F., Hofmann, K., Tjong Kim Sang, E., and de Rijke, M., editors (2013). *Cornetto: A Combinatorial Lexical Semantic Database for Dutch*. Springer.

Žabokrtský, Z. and Popel, M. (2009). Hidden Markov tree model in dependency-based machine translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 145–148, Stroudsburg, PA, USA. Association for Computational Linguistics.