Institute of Information and Communication Technologies
Bulgarian Academy of Sciences

Proceedings

of

The Workshop on Deep Language Processing for Quality Machine Translation (DeepLP4QMT)

Editors:
Kiril Simov
Petya Osenova
Jan Hajič
Hans Uszkoreit
António Branco

Supported by:

QTLeap Project

IICT-BAS

10 September 2016
Sofia, Bulgaria

The workshop is supported by:

The EC's FP7 under grant agreement number 610516: "QTLeap: Quality Translation by Deep Language Engineering Approaches."

Institute of Information and Communication Technologies,
Bulgarian Academy of Sciences

ISBN: 978-619-7320-03-9

Preface

In the last decade, research on language technology applications, such as machine translation (MT), information retrieval and extraction (also cross-lingual), etc. has benefited from the significant advances obtained with the exploitation of increasingly sophisticated statistical approaches. To a large extent, this advancement has been achieved also by encompassing a host of subsidiary and increasingly more fine-grained linguistic distinctions at the syntactic and semantic levels.

Thus, the NLP mainstream has headed towards the modeling of multilayered linguistic knowledge. To leap forward in terms of the quality of its output, machine translation and other technologies are taking advantage of enhanced capacities for deeper analysis of natural language and massive open online world knowledge that are now becoming available. The following initiatives can be mentioned as best practices, among others:

- LOGON MT system from-Norwegian-to-English which uses Minimal Recursion Semantics (MRS) and DELPH-IN deep HPSG grammar expertise for language transfer;
- Systems based on Abstract Meaning Representation (AMR);
- The ParGram parallel deep grammars and parsebanks covering several language families in the LFG formalism;
- The development of sophisticated syntactic and semantic models, sensitive to lexical semantics and semantic roles;
- Creation of high-quality parallel treebanks via model transfers (such as Prague Czech-English Dependency treebank);
- Creation of deep resources, such as English DeepBank, released in 2013;
- Creation of common tagsets and thus ‘universalizing’ linguistic resources, such as the Universal dependencies initiative, etc.

In the long run, richer world knowledge will be available, even beyond the current Linked Open Data, with respect to larger datasets, semantics enhanced with world facts, and more dynamic conceptual knowledge representation. Concomitantly, the evolutive trend in Natural Language Processing shows a strong integration of the knowledge-poor language processing with the knowledge-rich one, supported by deep grammars and deep language resources.

We would like to cordially thank all the involved colleagues: the organizers, the presenters and participants, the reviewers, and last but not least - our invited speakers. Enjoy the Proceedings!

Kiril Simov, Petya Osenova, Jan Hajič, Hans Uszkoreit, António Branco

Workshop Organization

Invited Speakers

Francis Bond, Nanyang Technological University
Josef van Genabith, DFKI

Program Chairs

Kiril Simov, IICT-BAS
Petya Osenova, IICT-BAS
Jan Hajič, Charles University in Prague
Hans Uszkoreit, DFKI
António Branco, University of Lisbon

Program Committee

Eneko Agirre, University of the Basque Country
Ondřej Bojar, Charles University
Gosse Bouma, University of Groningen
Aljoscha Burchardt, DFKI
Mauro Cettolo, FBK
Koenraad De Smedt, University of Bergen
Ondřej Dušek, Charles University
Markus Egg, Humboldt University of Berlin
Barbora Hladká, Charles University
Philipp Koehn, University of Edinburgh
Sandra Kübler, Indiana University
Gorka Labaka, University of the Basque Country
David Mareček, Charles University
Preslav Nakov, Qatar Computing Research Institute, Qatar Foundation
Stephan Oepen, University of Oslo
Martin Popel, Charles University
Rudolf Rosa, Charles University
Victoria Rosén, University of Bergen
João Silva, University of Lisbon
Inguna Skadiņa, Tilde Company and University of Latvia
Pavel Straňák, Charles University
Jörg Tiedemann, Uppsala University
Antonio Toral, Dublin City University
Gertjan van Noord, University of Groningen
Cristina Vertan, University of Hamburg
Dekai Wu, Hong Kong University of Science & Technology
Nianwe Xue, Brandeis University

Local Committee

Petya Osenova, IICT-BAS
Kiril Simov, IICT-BAS

Table of Contents

Choosing lemmas from Wordnet synsets in Abstract Dependency Trees *

Dieke Oele[1] and Gertjan van Noord[1]

Rijksuniversiteit Groningen, Groningen
d.oele@rug.nl, g.j.m.van.noord@rug.nl

Abstract. We explore lexical choice in Natural Language Generation (NLG) by implementing a model that uses both context and frequency information. Our model chooses a lemma given a WordNet synset in the abstract representations that are the input for generation. In order to find the correct lemma in its context, we map underspecified dependency trees to Hidden Markov Trees that take into account the probability of a lemma given its governing lemma, as well as the probability of a word sense given a lemma. A tree-modified Viterbi algorithm is then utilized to find the most probable hidden tree containing the most appropriate lemmas in the given context. Further processing ensures that the correct morphological realization for the given lemma is produced.

We evaluate our model by comparing it to a statistical transfer component in a Machine Translation system for English to Dutch. In this set-up, the word sense of words are determined in English analysis, and then our model is used to select the best Dutch lemma for the given word sense. In terms of BLEU score, our model outperforms a most frequent baseline, in which the most frequent lemma of a given word sense is always chosen. A manual evaluation confirms that our model is able to select the correct lemma when it is given a correct input synset. The majority of errors were caused by incorrect assignment of the word sense in the English analysis phase. Our model does not improve upon a transfer component trained on a parallel corpus. In the original transfer component, there barely are any lemmas that were incorrectly translated in the transfer phase, with the exception of Out of Vocabulary items (OOV's). In a further experiment we only used our model for OOV's and obtained a small improvement in BLEU score.

Keywords: Lexical choice, Generation, Tree-viterbi, HMTM

1 Introduction

This paper addresses the problem of lexical choice in Natural Language Generation (NLG). Lexical choice is a subtask of NLG, where an ideal model produces varied, natural-sounding, linguistic utterances. We consider generation systems that use abstract representations as input where the challenge lies in the construction of sentences on the basis of these representations.

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

The choice of a correct lemma is a difficult task which depends heavily on the quality of the dictionaries used. One such dictionary is WordNet [10], a lexical semantic database containing lemmas corresponding to their word meanings. Querying this database for a word returns a group of one or more synonyms called a synset containing a set of words of the same class. Being roughly synonymous in one of their meanings, makes them well suited for lexical choice.

Unfortunately, not every lemma in a synset is a full synonym of its original word which could cause errors when selecting the most probable variant without considering the context. Consider for instance the English WordNet synset: {"employment", "work"}. Both lemmas in this synset have the meaning of "The occupation for which you are paid". In the sentences of example 1 they are perfectly exchangeable. In example 2, however, both sentences require a different lemma in this particular context. This indicates that a more sophisticated system is required that selects a correct lemma given a synset while considering its context.

(1) 'He is looking for **employment**'
'He is looking for **work**'

(2) 'He is out of **employment***'
'He would like to terminate his **work*** agreement.

To this end, we propose the mapping of a dependency tree over synsets to a dependency tree over lemmas while taking into account both context information and 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. The latter makes them a good source for contextual information.

Our model takes as input directed labeled dependency trees with nodes corresponding to Wordnet synsets and its edges corresponding to syntactic relations. We use a Hidden Markov Tree Model (HMTM) and a Tree-Viterbi algorithm [4, 7, 22] to label the nodes of our dependency trees with a correct lemma by revealing the hidden states in the tree nodes, given another, observable, labeling of the nodes of the same tree.

The independence assumptions that are made by HMTMs can be useful for modeling dependency trees. They fit dependency trees well, since they assume conditional dependence only along the tree edges, which corresponds to intuition behind the linguistic dependency relations in dependency trees. Moreover, HMTMs can be used for the labeling of nodes in a dependency tree. In our model this, can be interpreted as the revealing of the hidden states (the lemmas) in the tree nodes, given another observable labeling of the nodes of the same tree (the synsets) by use of a tree-modified Viterbi algorithm. The resulting labeled dependency trees should then comprise the correct lemmas given their context and can be used to as input to a generation system.

This paper is structured as follows. Section 2, describes some previous work regarding lexical choice and HMTM. In section 3, we introduce the HMTM model and the Tree-Viterbi algorithm for lexical choice. Section 4 gives a description of experiments that test the model. The results of the experiments are shown in section 5 and in section 6 we discuss the obtained results and suggest some improvements.

2 Previous work

The problem of lexical choice is not broadly considered in previous work, 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) investigated the criteria for a good lexical choice in NLG and marked semantic context as the most important one. Yet, another approach to lexical choice is the consideration of conversational aspects of the problem such as pragmatics. Hovy (2013), for example, proposes to address the selection of lexical items according to rhetorical goals of the speaker that influence stylistic goals and therefore lexical choices. On the other hand Elhadad (1992) focuses on the influence of the speaker's argumentative intent on lexical choice. Wanner and Bateman (1990) consider the degree of salience of semantic elements and regard lexical choice as a situation dependent aspect. They claim that aspects of a message to be expressed by a generator can have different degrees of salience, which may influence lexical choice. A number of algorithms and models have been developed for lexical choice, for example Bangalore and Rambow (2000) investigated different tree-based stochastic models for lexical choice that relied on corpus information and 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 [12]. An example of such a system is Basile (2014) who proposes an unsupervised algorithm for lexical choice from WordNet synsets that exploits the WordNet hierarchy of hypernyms and 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 model are: image segmentation, signal classification, denoising and image document categorization [7]. The use of the model in natural language processing is fairly new and has been applied to word alignment [13] and Machine Translation [22]. Žabokrtský and Popel (2009) were 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.

3 Method

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

3.1 Hidden Markov Tree Model for lexical choice

HMTMs are similar to the well known Hidden Markov Models (HMM) as they both contain a sequence of observed states with corresponding hidden states [6, 7]. However, instead of a linear chain of observations, they map over a tree of observations. Furthermore, analogously to regular HMMs, HMTMs rely on emission probabilities and transition probabilities. In the Markov process for lexical choice, we assume that we are given a directed labeled dependency tree. Its nodes correspond to synsets, used for emission probabilities, and its edges correspond to syntactic-semantic relations, used for transition probabilities.

The hidden states in the tree nodes are revealed on the basis of an observed labeling of the nodes of the same tree. When using HMTMs for lexical choice, the hidden states consist of actual lemmas, whereas the observations are word senses (synsets). Figure 1 contains an example of an HMTM containing synsets for a Dutch sentence. In this particular context from the synset $\{'poeder','pulver','stof','poeier'\}$, the best choice of lemma would be "stof", and it is up to the tree-viterbi algorithm to choose this option over the other lemmas, given the synset and its context.

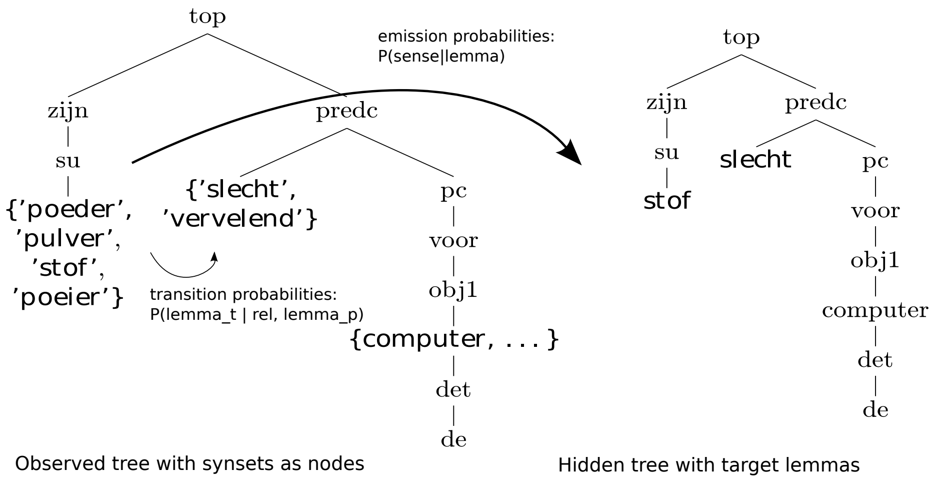


Fig. 1. HMTM for the Dutch sentence: *"'***Stof** *is slecht voor de computer"* ("Dust is bad for your computer")

The tree is defined by an observed dependency tree containing synsets as nodes, $S = \{S(n_1), ..., S(n_m)\}$, and a hidden tree with target lemmas, $T = \{T(n_1), ..., T(n_m)\}$, isomorphic to the observed tree where m is the size of the tree. The function $\pi(n)$ returns a pair (n', l) where n' is the unique parent of node n (with r corresponding to the root of the tree), and l is a label indicating the nature of the dependency. Such labels include *subject, object, modifier,* $\ldots$. Each node, except the root node, refers to a word in the sentence. Like HMMs, HMTMs make two independence assumptions: given $T(\pi(n))$, $T(n)$ is conditionally independent of other nodes and given $T(n)$, $S(n)$ is conditionally independent of other nodes.

The required frequency counts for the emission probabilities of a sense given a target lemma can be obtained from sense annotated corpora or from the output of WSD-systems. We need to estimate the probability of an observed state (the sense), given the hidden state (the lemma):

$$P(sense|lemma) \approx \frac{freq(sense, lemma)}{freq(lemma)} \tag{1}$$

Consider for instance the probability of the synset $\{lager, beer, ale, ...\}$ given the lemma "beer". If the lemma "beer" is associated in the corpus with the $\{lager, beer, ale, ...\}$ sense in 89 out of a 100 cases, then the emission probability will be estimated as 0.89.

The transition probabilities of a target lemma $lemma_t$ given a dependency relation rel (such as subject, object or modifier) and its parent $lemma_p$, can be collected from large parsed corpora. For this we can use the following equation:

$$P\left(lemma_t \mid rel, lemma_p\right) \approx \frac{freq(lemma_p, rel, lemma_t)}{freq(lemma_p, rel)} \tag{2}$$

If we want the probability of the lemma "beer" given a parent "drink" in the dependency relation "obj" we compute:

$$P(t_i|t_{i-1}) = p\left(beer \mid obj1, drink\right) = \frac{freq(drink, obj1, beer)}{freq(drink, obj)} \tag{3}$$

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.

3.2 Tree-Viterbi

The most probable hidden tree labeling given the observed tree labeling can then be found by use of a modification of the traditional Viterbi algorithm for HMMs to a Tree-Viterbi algorithm for HMTMs. Details on this modification can be found in Durand et al. (2004) and Diligenti et al. (2003). The tree-viterbi algorithm, 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.

4 Experiments

Although our method can be applied to various NLG problems, such as paraphrasing or summarization, the evaluation of our model requires dependency trees containing synsets as nodes. We could obtain them by using a dependency parser to parse sentences in a monolingual setting, use the resulting trees as input to our model and subsequently generate sentences. Instead, we test our model in a transfer-based Machine Translation (MT) system. The advantage of this setup is that we have more realistic data as compared to the monolingual setup. We can use the dependency trees that are the result of the transfer phase in the MT-pipeline and apply the Tree-Viterbi algorithm before generation.

Our model takes abstract dependency structures over senses as input which can easily be obtained by applying word sense disambiguation (WSD) on the source side of the pipeline, store them in the nodes of the dependency tree and retain them during transfer. For this, we used the UKB-WSD module [1] in the English analysis phase of the MT pipeline resulting in dependency trees containing synsets in their nodes. The English synsets are then converted to Dutch synsets from the Cornetto database [21]. For each node in the dependency tree that contains a synset, our model is used to find an optimal lemma given its synset and its context. This lemma is then used to substitute the one that was originally translated by the translation model in the transfer phase. Ultimately, the trees are used to generate full sentences.

For the experiments, the answers-part of Batch 1 of the QTLeap corpus [14] were used. This data set consists of an IT help-desk scenario that contains translations of customer data into each of the QTLeap project languages. In addition, the model was tested on a broader domain test set. For this, we took 1000 sentences from the English-Dutch News Commentary data set [17].

The data is analyzed and translated from English to Dutch with Treex, a tree-to-tree machine translation system whose translation process follows the analysis-transfer-synthesis pipeline [15, 23]. The sentences are analyzed and translated from English to Dutch with Treex, a modular framework for natural language processing [15]. It contains a tree-to-tree machine translation system whose translation process follows the analysis-transfer-synthesis pipeline [22]. In the analysis phase, a source sentence is transformed into a deep syntax dependency representation. 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 representations that are the input for the generation of Dutch sentences. In the analysis phase, a source sentence is transformed into a deep syntax dependency representation. The resulting dependency trees are transferred to Dutch abstract representations that are the input for the generation of Dutch sentences.

For generation we use the the Alpino Generator [5, 18] that generates Dutch sentences on the basis of an abstraction of dependency structures. The Alpino system for Dutch [18] 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) [5].

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 [5], 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.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). Similar to normal dependency trees, they consist of a syntactical representation of a sentence in the form of a tree. The generation process starts with an ADT and then uses the grammar to construct one or more sentences. Ultimately, a statistical model is used for choosing the most fluent one.

The emission probabilities are taken from DutchSemCor [20]. An important issue, however, is that the sense-annotated corpus only contains counts for a limited number of lemma, which can be problematic when estimating emission probabilities. If the target lemma does not appear in the corpus, it is not considered, possibly causing a less suitable lemma to be chosen. In our data, for example, for a synset containing the following senses: *{'bladzijde', 'pagina', 'zijde'}* the lemma "pagina" will not be chosen as it does not appear in our sense-tagged corpus. We therefore applied a simple heuristic on these "missing" lemmas to estimate the probability of the sense by dividing 1 by the number of synsets the lemma appears in. Transition probabilities are obtained from a large set of parsed corpora. For this we take the SONAR part of Lassy Large [19], containing approximately 500M words. From these parses, dependency relations and their counts are retrieved resulting in a transition probability matrix that can be queried for each lemma given its parent lemma and their relation.

5 Results

In WSD, a typical baseline consists of taking the most frequent sense of the target word. We adopt the idea behind this baseline for the lexical choice problem by looking at the frequency distribution of a lemma/synset combination in DutchSemCor. As can be seen in table 1, the Tree-Viterbi algorithm performs better compared to the most frequent baseline. This is confirmed by a manual comparison of the lemmas chosen by our model with the ones picked by the baseline system. For this evaluation we took a random subset of 100 lemmas, chosen by our model, and assessed whether they where either correct or incorrect choices. If the choice of lemma was incorrect it was determined whether this was either caused by the model itself or by the fact that a wrong WordNet synset was chosen as input. From these 100 choices, 56 where correct lexical choices. Of the remaining 44 erroneous choices, 33 are caused by a wrong input sense causing the output to be worse compared to the original MT output.

Table 1. BLEU scores for English-Dutch translation using the tree-viterbi algorithm

	QTLeap	News
Most frequent	20.16	07.00
Tree-viterbi	21.93	07.45

Our manual evaluations shows that the amount of wrong input senses is fairly large. However, when our model does get a correct input sense, it makes a better choice in most cases. In the output, examples can be found where the tree-viterbi algorithm outperforms the baseline. In example 3, for instance, the baseline system chooses the incorrect adjective "bezig" (busy) instead of "actief" (active). Example 4 demonstrates that the inclusion of dependency information of the tree-viterbi algorithm works well. The most frequent lemmas that replace “harde schijf” (hard disc) with “sterke bedrag” (strong amount) are avoided by our model, even though the wrong synset was chosen as input. For the replacement of ’schijf’ the system can choose between “bedrag” (amount), “schijf” (disc) and “som” (sum), with the latter having the highest emission probability. For the lemma “hard” it can choose between multiple adjectives, including “sterk”. The transition probabilities of the combination of both lemmas ensures that the right lemmas are chosen by the algorithm.

(3) *English original:* Access your friend’s profile to see if the account is still **active**.
Most frequent: Open je vriend profiel om te kijken of het account nog **bezig** (“busy”) is.
Tree-viterbi: Open je vriend profiel om te kijken of het account nog **actief** (“active”) is.

(4) *English original:* Your **hard drive** has two USB configurations.
Most frequent: Je **sterke bedrag** (“strong amount”) heeft twee USB uitvoeringen.
Tree-Viterbi: Je **harde schijf** (“hard drive)” heeft twee USB uitvoeringen.

5.1 Tree-Viterbi for out-of-vocabulary items

Since we evaluate our system in an MT-setup, we can also compare it with the original MT-output that does not use the Tree-Viterbi algorithm for lexical choice. Our model appears to make an improvement in particular in cases where the original language model was not able to find a translation and therefore yields the original, non-translated, word. Using the algorithm for lexical choice only on these out-of-vocabulary items (OOV’s) could thus improve the output.

Table 2. BLEU scores for English-Dutch translation using the tree-viterbi algorithm only on OOV’s

	QTLeap	News
Original MT-output	23.02	08.52
Tree-viterbi for OOV	23.03	08.63

In table 2, the results can be found of a second experiment on OOV’s. The BLEU scores for both test sets are similar to the scores of the system without lexical choice. However, when looking at the output manually, the algorithm does improve the original sentences in almost all cases, yielding more fluent and natural sounding sentences. Examples of these improvements can be seen in sentence 5 and 6. Since OOV’s are not very common in the test data, our model finds a Dutch word for 132 of them, which could explain the small difference in BLEU score. From a random subset of these lexical choices, it was manually assessed that 7 of them are wrong choices made by our model, 33 of them are errors caused by a wrong input sense, while 60 are good lexical choices.

(5) Zet je wachtwoord in **[rectangle => rechthoek]** en klik op log In.
Insert your password in the rectangle and click Log In

(6) Probeer de belangrijke combinatie van **[brightness => helderheid]** te controleren.
Try to check the important combination of brightness

6 Discussion

From our evaluation it becomes clear that the model is able to select the correct lemma if the correct input synset is available. However, it also becomes apparent that the algorithm for lexical choice is not very suitable in an MT-Setup. Although the HMTM outperforms the most frequent baseline, the MT-system does not improve when the algorithm is used on OOVs. More importantly, since the systems are trained on domain-specific data which probably decreases the chance of translating a lemma incorrectly in the transfer-phase, they are rarely incorrect. This leaves little room for improving the output by way of lexical choice, while still having the same chance of causing errors.

The main cause of errors, therefore, are wrong input senses. In most of the cases where the tree-viterbi model chooses a wrong lemma, an incorrect synset was used as input. A target lemma cannot be retrieved from a source synset in some context if the input synset appears in a different sense than the one in which it is synonymous with the target. When a wrong synset is chosen as input, the system has a high chance of selecting a wrong lemma. Consider for example the lemma "menu", that appears in two Dutch synsets:

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

Since the data used for the experiments belongs to the IT domain, the second synset, in bold, is the preferred one. When first synset, with the meaning of *restaurant menu* would be used as input to the algorithm the lemma "kaart" (map) could be chosen in stead of "menu" which is usually not preferred in this domain.

In this MT-setup, where the transfer model is trained on domain specific data, our system is, not able to significantly improve the translation output. Since the systems are trained on domain-specific data which probably decreases the chance of translating a lemma incorrectly in the transfer-phase, they are seldom wrong. This leaves little room for improving the output by way of lexical choice, while still having the same chance of causing errors. However, the fact remains that our model is able to yield a similar score without the use of domain specific parallel data and could therefore be useful in settings where this data is not available.

Another problem that is highly likely to cause errors in this setup 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.

7 Conclusion

In this work we intended to tackle the problem of lexical choice by using HMTMs. 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. To evaluate the algorithm, it was implemented in a Machine Translation system. The results of our experiments indicate that the algorithm for lexical choice on the basis of an HMTM works very well. In terms of BLEU score, our model outperforms the most frequent baseline. Also, when manually evaluating the output, it becomes clear that our model chooses better lemmas. When using the algorithm only on OOV-items, in terms of Bleu score, the system yields similar scores as compared to the one that does not use lexical choice in the generation phase without the use of domain specific parallel data.

Our model for lexical choice takes notion of the combination context and frequency information and therefore contributes to the generation of fluent natural sounding, sentences.

Bibliography

[1] Eneko Agirre and Aitor Soroa. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 33–41, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1609067.1609070`.

[2] Srinivas Bangalore and Owen Rambow. Corpus-based lexical choice in natural language generation. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 464–471, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075277. URL `http://dx.doi.org/10.3115/1075218.1075277`.

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

[4] Matthew S Crouse, Richard G Baraniuk, and Robert D Nowak. 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, 1996.

[5] D.J.A. De Kok. *Reversible Stochastic Attribute-value Grammars*. Groningen dissertations in linguistics. 2013. ISBN 9789036761123. URL `http://books.google.nl/books?id=uNlcmwEACAAJ`.

[6] Michelangelo Diligenti, Paolo Frasconi, and Marco Gori. Hidden tree Markov Models for Document Image Classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(4):519–523, 2003.

[7] Jean-Baptiste Durand, Paulo Goncalves, and Yann Guédon. Computational Methods for Hidden Markov Tree Models-An Application to Wavelet Trees. *Signal Processing, IEEE Transactions on*, 52(9):2551–2560, 2004.

[8] Philip Edmonds and Graeme Hirst. Near-synonymy and lexical choice. *Comput. Linguist.*, 28(2):105–144, June 2002. ISSN 0891-2017. doi: 10.1162/089120102760173625. URL `http://dx.doi.org/10.1162/089120102760173625`.

[9] Michael Elhadad. *Using argumentation to control lexical choice: a functional unification-based approach*. Phd dissertation, Department of Computer Science, Columbia University, 1992.

[10] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, MA ; London, May 1998. ISBN 978-0-262-06197-1.

[11] Eduard Hovy. *Generating Natural Language Under Pragmatic Constraints*. Culture and Communication in Asia. Taylor & Francis, 2013. ISBN 9781134742219. URL `https://books.google.ch/books?id=lbDDajegkP8C`.

[12] Hongyan Jing. Applying wordnet to natural language generation. In *University of Montreal*. Citeseer, 1998.

[13] Shuhei Kondo, Kevin Duh, and Yuji Matsumoto. Hidden markov tree model for word alignment. In *Proceedings of the Eighth Workshop on Statistical Machine*

Translation, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[14] Petya Osenova, Rosa Del Gaudio, João Silva, Aljoscha Burchardt, Martin Popel, Gertjan van Noord, Dieke Oele, and Gorka Labaka. Interim Report on the Curation of Language Resources and Tools for Deep MT. Technical Report Deliverable D2.5, Version 2.0, QTLeap Project, 2015.

[15] Martin Popel and Zdeněk Žabokrtský. Tectomt: Modular nlp framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing*, IceTAL'10, pages 293–304, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-14769-0, 978-3-642-14769-2. URL `http://dl.acm.org/citation.cfm?id=1884371.1884406`.

[16] Manfred Stede. 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, 1993.

[17] Jörg Tiedemann. Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association, 2012.

[18] Gertjan van Noord. **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, 2006.

[19] Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. *Large Scale Syntactic Annotation of Written Dutch: Lassy*, pages 147–164. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-30910-6. doi: 10.1007/978-3-642-30910-6_9. URL `http://dx.doi.org/10.1007/978-3-642-30910-6_9`.

[20] Piek Vossen, Attila Görög, Rubén Izquierdo, and Antal Van den Bosch. DutchSemCor: Targeting the Ideal Sense-tagged Corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7.

[21] P.T.J.M. Vossen, H. van der Vliet, I. Maks, R. Segers, M.-F. Moens, K. Hofmann, E.F. Tjong Kim Sang, and Maarten de Rijke, editors. *Cornetto: A Combinatorial Lexical Semantic Database for Dutch*. Springer, 03/2013 2013.

[22] Zdeněk Žabokrtský and Martin Popel. 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, 2009. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1667583.1667628`.

[23] Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. TectoMT: Highly Modular MT System with Tectogrammatics Used As Transfer Layer. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 167–170, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

ISBN 978-1-932432-09-1. URL http://dl.acm.org/citation.cfm?id=1626394.1626419.

[24] L. Wanner and J. A. Bateman. A collocational based approach to salience-sensitive lexical selection. In *Proc. of the Fifth International Workshop on Natural Language Generation*, pages 31–38, Dawson, PA, 1990.

Language Resources for Deep Machine Translation

Petya Osenova, Stanslava Kancheva, Svetlomira Manova, Ivajlo Radev, and Nadezhda Terzijska

Institute of Information and Communication Technologies, BAS,
Akad. G. Bonchev. 25A, 1113 Sofia, Bulgaria
{petya,stanislava,svetlomira,ivajlo,nadya}@bultreebank.org

Abstract. The paper discusses various types of Language Resources that contribute to Deep Machine Translation. These resources comprise lexicons (such as wordnets, valency lexicons and DBpedia) and corpora (such as semantically annotated treebanks), both monolingual and bilingual. In this paper the situation for Bulgarian is presented, but the described ideas are applicable also to other languages.

Keywords: Language Resources, Deep Machine Translation, Bulgarian, lexicons, treebanks

1 Introduction

Deep Machine Translation (DMT) is usually supported by two types of language resources (LRs): (1) Treebanks as syntactically annotated corpora and (2) Lexicons as providers of semantic and valency information. Treebanks have been used for training deep processing tools as well as deep (tree-based) MT models, while lexicons enhance the deep language processing with terminological, grammatical and sense information.

In this paper the language resources for Bulgarian are decsribed as a source and target language in DMT. These LRs are meant to enhance deep processing for the aims of DMT.

The following LRs are considered:

1. A semantically annotated treebank
2. A valency Lexicon for Bulgarian verbs
3. A Bulgarian WordNet (BTB-WordNet)

All of these resources are outlined from the perspective of the semantic annotation process.

The paper is structured as follows: the next section describes the overall architecture of the LRs. Section 3 presents the monolingual LRs. Section 4 outlines the bilingual LRs. Section 5 concludes the paper.

2 The architecture of Language Resources

The overall architecture of the LRs contribution in annotation is presented in Fig. 1. Note that the syntactic information is not presented explicitly, but it is assumed to be there. Let us start the explanation from the bottom to the top. The words in the text are signified by the letter *W* with the appropriate word position number - 1, 2, i. The named entities are signified by the label NE. From the BTB-Wordnet thesaurus the words receive their meanings in the context. At the same time the concepts from DBPedia Ontology are mapped to the concepts in BTB-Wordnet. The DBPedia Ontology is related to the DBPedia instances through the relation 'instance' and to the NE Annotation block through the relation 'classification'. According to our strategy, when there is no DBPedia instance, related to the NE type with th relation URI, this NE type maps only to the concept in the DBPedia Ontology. The LRs are either monolingual (Bulgarian) or bilingual/multilingual. In our case the bilingual resources refer to Bulgarian and English.

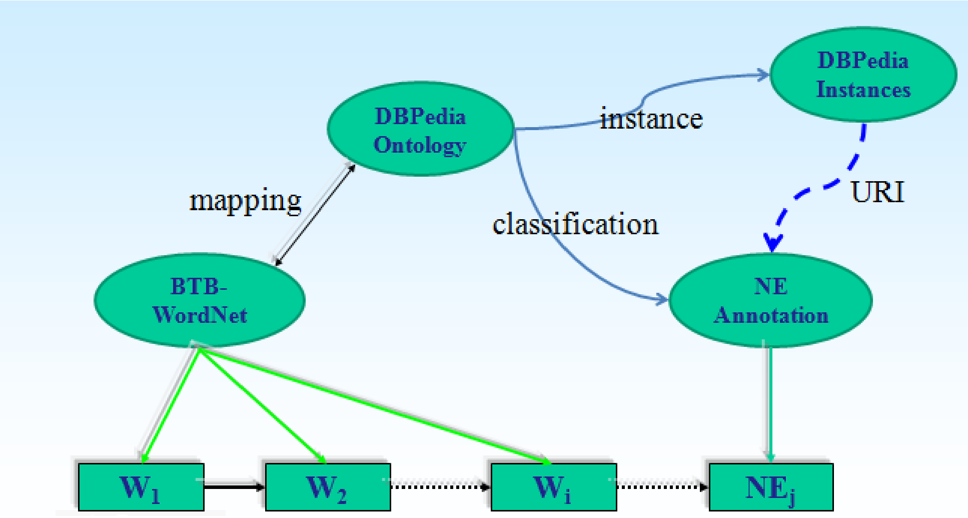


Fig. 1. The architecture of relations among Language Resources.

3 Monolingual language resources

The canonical sense-annotated corpus is SemCor and its variants. There are not so many treebanks that have been annotated with senses. These are, for example, PropBank, OntoNotes, TÜBa-DZ [1]; the Italian syntactic-semantic treebank [2] and the Polish treebank [3]. The novelty in our sense annotation endeavour is the combination of the assigned valencies, the lexical senses and the DBpedia URIs into a syntactically annotated language resource – BulTreeBank.

The original HPSG resource comprises 256,331 tokens, which form a little more than 15,000 sentences. Its first conversion was performed in 2006, when it was transferred into the shared CoNLL dependency format (18 relations, 196 000 tokens). Now is has been converted also into the UD format (156 000 tokens)[1].

The sense annotation covers the open class words in BulTreeBank [4]. Three groups of items have been annotated: common words, MultiWord Expressions (MWE) and Named Entities. The semantic information was selected from four sources: BTB-Valency lexicon; BTB-WordNet; Bulgarian DBpedia instances (Wiki pedia pages) and DBpedia ontology.

The syntactic structure of the treebank was used for extracting of new relations between semantic units (synsets, DBpedia instances and classes). These relations have been used as semantic restrictions over the valency frames in a valency lexicon for Bulgarian. Also, the extracted relations have been used for the task of the Knowledge-based Word Sense disambiguation.

The sense annotation strategy comprizes the following steps:

1. Mapping the lemma of the existing synsets or definitions in the dictionary to the lemma of the common word in the treebank.
2. Selection of one of the synsets or one of the definitions.
3. Defining a mapping to the Princeton WordNet in case of non-mapped definitions.
4. Detecting other usages of the same common word in the treebank.
5. Forming a new synset in the BTB-WN.
6. Returning its internal identifier to the sense annotation in the treebank.

The DBPedia annotation covers 10 885 named entities among which 2877 organizations, 2938 locations, 4195 people (the rest were from other different categories: events, books, others). The gazetteers for the DBpedia instances were created automatically from a dump of the Bulgarian DBpedia. All the Named Entities in the treebank were annotated with all the possible URIs for the DBpedia instances. The annotator selected an appropriate URI for the NE. In case when there was no appropriate DBpedia instance, the annotator tried to find an appropriate Wikipedia page. An example is given on Fig.2. The gloss is: ‘Travelled.HE for village Birimirtsi’. The translation is: He had been travelling to the village of Birimirtsi.

The valency annotation has the following steps:

1. Valency frames have been assigned by a verb valency lexicon of Bulgarian.
2. Each verb was annotated with the appropriate sense in the context.
3. The valency frames were assigned to the senses of the verbs.

In case of ambiguity, the semantic restrictions over the arguments have been used.

[1] http://universaldependencies.org/#bg

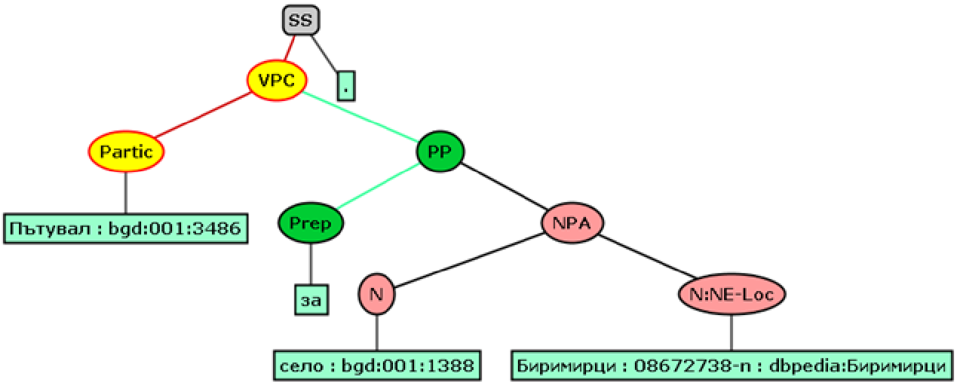


Fig. 2. A sentence annotated with information from DBPedia.

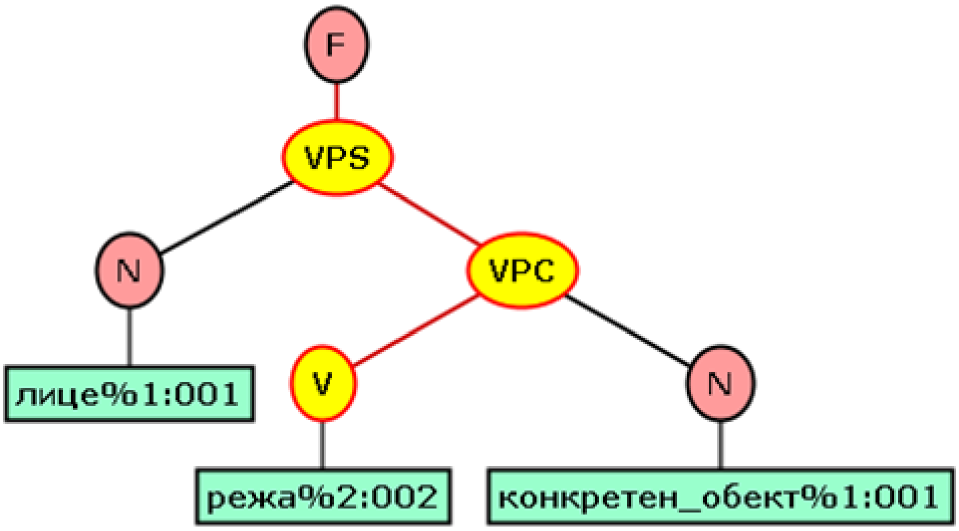


Fig. 3. A Valence Frame.

An example of this type of annotation is given on Fig. 3. The gloss as well as the translation is: ‘person cut concrete object’. The frame already shows the ontological generalization over the frame participants.

4 Bilingual language resources

The bilingual resources that we consider appropriate for DMT are of two types: parallel treebanks and thesauri, such as wordnet. Let us consider them in brief here.

Parallel treebanks: BulEngTreebank consists of 920 sentences (9308 tokens) in the domain of tourism. It has been analysed by the English Resource Grammar (ERG), and then – manually disambiguated. The sentences were translated into Bulgarian by professional translators. The Bulgarian and English sentences have been aligned manually on the word level. Then they were annotated morphologically and parsed by a dependency parser. The result was manually corrected.

ParDeepBankBG consists of 838 sentences (21 949 tokens) from the Bulgarian English Parallel Deepbank. The domain is journalism/finance (Wall Street Journal) The texts have been aligned manually on the word level and partially checked.

The QTLeap corpus consists of IT helpdesk questions and answers (4 batches - 1000 interactions each). The Bulgarian translations have been checked, cleaned and adjusted wrt to: tokenization; Latin vs. Cyrillic wording; errors; comprehensibility (ellipses, etc.).

Wordnet BTB-WN has been mapped to the PWN. The types of mappings follow:

1. full correspondence (one-to-one)
2. partial correspondence (one-to-many or many-to-one)
3. forced connectivity (re-design of the Bulgarian definition)
4. resolving metonymies, incorrect and extended correspondences.

The idea behind the full correspondence mapping in its ideal version is when the concepts in the two languages map one-to-one (Bulgarian *sigurnost* and English *certainty* that both mean ‘lack of danger’). In the partial correspondence case the Bulgarian concept is more specific than the English one. Then, it is mapped to a more general English one with a subsumption relation (*uncle* as mother’s brother is *vujcho* in Bulgarian, while *uncle* as father’s brother is *chicho* in Bulgarian).

On the other hand, the Bulgarian concept can be more general and subsume one or more concepts from PWN. In this case, the Bulgarian concept is mapped to each of the more specific English ones with the specificity relation. For example, *Mafia* in Bulgarian as criminal organization was mapped simultaneously to *Cosa Nostra* and to *Sicilian Mafia* in PWN. In our efforts to achieve one-to-one mapping for concepts with a disjunctive definition, the Bulgarian concept often had to be divided into two more specific ones.

5 Conclusion

Deep Machine Translation is enhanced by the availability of two main types of LRs: corpora and lexicons. It is a well known fact that the richer they are, the better; the more multilingual they are, the better.

The lexicons can refer to lexical databases such as wordnets, to valency dictionaries and Linked Open Data repositories (such as DBPedia, GeoNames, etc.).

The corpora can refer to semantically annotated treebanks. Semantics might include word meanings, logical forms, semantic roles, coreferences, etc.

At the same time, we should be aware of the problematic issues when exploiting such resources. The main issues are: existing errors; missing information; sparseness. In the bi- and multilingual settings very often the LRs for one of the languages are not rich and big enough; also the construction or annotation approaches might differ, which would make their joint usage difficult.

Last but not least, we should mention the time-consuming methods of quality resource construction as well as the across-domain applications.

Thus a conclusion can be made that the availability of rich LRs is a must for DMT, but important is also the way in which they are involved and the way they in which they are combined.

Acknowledgements

This research has received partial support from the EC's FP7 project: "QTLeap: Quality Translation by Deep Language Engineering Approaches" (610516).

References

1. Henrich, V., Hinrichs, E.: Extending the tÜba-d/z treebank with germanet sense annotation. In: Iryna Gurevych, Chris Biemann, and Torsten Zesch (eds.): Language Processing and Knowledge in the Web, Lecture Notes in Computer Science. (2013) 89–96
2. Montemagni, S., Barsotti, F., Battista, M., Calzolari, N., Corazzari, O., Pirrelli, A.L.V., Zampolli, A., Fanciulli, F., Massetani, M., Raffaelli, R., Basili, R., Pazienza, M.T., Saracino, D., Zanzotto, F., Mana, N., Pianesi, F., Delmonte, R.: The syntactic-semantic treebank of italian. an overview. In: Linguistica Computazionale XVI-XVI. (2003) 461–492
3. Hajnicz, E.: The procedure of lexico-semantic annotation of składnica treebank. In: Proceedings of LREC-2014. (2014) 2290–2297
4. Popov, A., Kancheva, S., Manova, S., Radev, I., Simov, K., Osenova, P.: The sense annotation of bultreebank. In: Proceedings of TLT13. (2014) 127–136

Neural Network Language Models – an Overview

Alexander Popov

Institute of Information and Communication Technologies, BAS,
Akad. G. Bonchev. 25A, 1113 Sofia, Bulgaria
alex.popov@bultreebank.org

Abstract. This article presents a brief overview of different approaches to training language models for natural language processing, using neural networks. It contrasts the advantages and disadvantages of traditional count-based approaches and newer, neural ones. Then it presents the evolution of neural network language models, starting with simple feed-forward architectures and going to recurrent ones, including their more complicated LSTM variant. It also briefly comments on the relation between word embeddings and neural network language models.

Keywords: Language Models, Feedforward Neural Networks, Recurrent Neural Networks, Long Short-Term Memory, Word Embeddings

1 Introduction

Language models (LM) are crucial components in many systems that solve NLP problems. An LM is used to predict the probability of generating a particular sequence of words, based on previous observation of text. LMs are used in various NLP applications, such as speech recognition, spelling correction, part-of-speech tagging, machine translation.

For a long time, the most successful techniques for language modeling have been based on counting the words in large corpora. Such models are trained by obtaining maximum likelihood estimations of the occurrence of particular word sequences in text. That is, n-gram occurrence frequencies are calculated against the occurrences of *(n-1)*-grams (or against all the words in the case of unigrams).

Count-based LMs have obvious advantages, mainly due to their relative simplicity, which permits us to think about them very clearly. They are fully generative, in the sense that they can generate whole, observable sequences, as well as estimate the probabilities of observed ones. The algorithms used for training models like that and those employed for decoding the most probable sequences are also well-understood.

However, count-based LMs also suffer from a number of disadvantages. They are predicated upon very strong independence assumptions (also called Markov assumptions) and as such do not constitute a very accurate linguistic representation. Count-based LMs can be made more precise by increasing the length of n-grams, but as N grows, data becomes increasingly sparser (*5*-grams are a typical choice for good results). Training is usually done on corpora with millions of words, but even then n-grams of higher orders are very rare in the data.

Smoothing techniques are applied to alleviate this problem; however, this is a difficult task in itself. Unknown words at test time present a further challenge.

Yet, despite all obstacles, until recently count-based methods have been the predominant paradigm in language modeling. It is only some years ago that neural network LMs came to challenge them, producing results that compete and surpass previous achievements.

Neural networks (NNs) bring together with them several significant advantages over backoff n-gram models. It is easy and natural to adapt NN architectures in order to project input words into lower-dimensional representations (e.g. dimensions of a size of 500, wherein words are represented as 500-positional vectors). This allows for the automatic clustering of similar words in the embedding space (word representations in such a space are now popular in the literature as *word embeddings*). Moreover, certain lexico-grammatical properties of the words are encoded along the dimensions of these distributed representations. This effectively solves the problem with unseen sequences, as words are in this way quickly clustered around other results that have been seen to occur in similar positions. This equates in practice to implicitly solving the smoothing problem.

There are other strengths that NNs possess in relation to NLP tasks. For instance, unknown words in LMs can be handled via NNs by building word representations that are constructed from embeddings based on morphological elements or even characters, rather than directly from merely tokenized data (for character-based word embeddings see [1]; for suffix embedding see [2]). NNs also permit the resolution of the long sequence problem, which is achieved by the addition of recurrent links between the hidden layers of the networks. Below is given a brief overview of the different ways language modeling can be done with NNs.

2 Feedforward Neural Network Language Models

One of the simpler ways to do language modeling using NNs is to adapt *feedforward neural networks* to the task. Feedforward NNs are in some ways similar to count-based LMs, because context is still represented as a sliding window of *N-1* words. Thus, at each time step a word is examined in its fixed context, then the window moves and the next word is examined in much the same way as analysis is performed on n-grams in count-based LMs. Nevertheless, feedforward NNs have the advantage over older methods of projecting the input words into lower-dimensional space.

Figure 1 shows a schematic representation of a feedforward LM. All words in the sliding window context are embedded in the shared space and then their vectors are concatenated and fed into a hidden layer that compresses the context for the current word. One final transformation (typically a softmax activation function outputting a probability distribution) unpacks that representation into a vector that has as many positions as there are words in the vocabulary. The word whose position has the highest probability associated with it is selected as the final output. Large output vocabularies sometimes present a problem, as

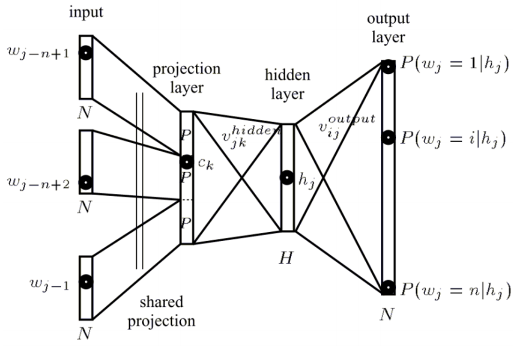


Fig. 1: Feedforward neural network language model; figure taken from [3].

this reflects on the size of the matrix computation at the final step and can considerably slow down training. One way to alleviate this issue is to use a *hierarchical softmax*, i.e. to decompose the final probability into two factors: the probability of the word belonging to a particular class and the probability of it being a specific word from that same class (see [4]). If the inventory of classes is selected well, this can reduce the computational complexity significantly.

3 Recurrent Neural Network Language Models

The next step in making NNLMs more sophisticated is to enable them to keep an indefinitely long history of the previously observed words in a sequence. This is accomplished by making the networks *recurrent*, i.e. a link is added between the hidden layer (or layers) and itself, which allows the layer to keep a kind of dynamic memory of what has passed through the network as input at previous time steps (see fig. 2).

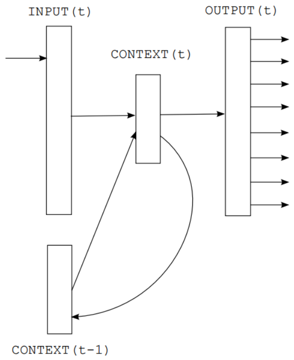


Fig. 2: Schematic representation of a recurrent neural network language model; figure taken from [3].

The hidden layer with recurrent connections is often called the *state* of the network and is calculated by running the concatenation of the input word vector and the previous state (*s-1*) through an activation function, e.g. the sigmoid function. Finally, the state is fed into a softmax activation function to produce a probability distribution and thus select the most probable word (again, a hierarchical factorization can be introduced to speed up training). This simple addition (which is essentially captured by an extra matrix that holds the recurrent links and the concatenation of the state vector to the input at each time step) allows RNNs to process different sequences, rather than to deal merely with snapshots of fixed lengths.

4 Long Short-Term Memory Network Language Models

RNNs are a powerful tool for working with sequential inputs, but they do suffer from a major setback in their simplest incarnation. Neural networks are trained through the backpropagation algorithm which calculates the error gradients at different time steps and returns that value to the lower layers, which are then updated based on that feedback. This works well for shorter sequences, but when longer dependencies obtain in the input, gradient calculation can quickly explode or vanish, i.e. the gradients become too big or too close to 0. This is known as *the exploding/vanishing gradients problem* and it has hampered the serious use of RNNs for a long time, until the invention of mechanisms to neutralize it. One such mechanism (along with *Gated Recurrent Units* [5]) are Long Short-Term Memory cells, introduced by [6].

The module in the plain RNN that is looped to itself has a single hidden layer inside; an *LSTM block* is internally more complex. Its cell state is modified by the outputs of several gates: *forget gate*, *input gate* and *output gate*. The LSTM cell has four hidden layers inside, which learn the appropriate weights needed to correctly forget and update (parts of) the cell state. Their outputs are calculated via the following equations ([7]):

$$\mathbf{i_t} = \sigma(W_{\mathrm{xi}}x_{\mathrm{t}} + W_{\mathrm{hi}}h_{\mathrm{t\text{-}1}} + W_{\mathrm{ci}}c_{\mathrm{t\text{-}1}} + b_{\mathrm{i}}) \tag{1}$$

$$\mathbf{f_t} = \sigma(W_{\mathrm{xf}}x_{\mathrm{t}} + W_{\mathrm{hf}}h_{\mathrm{t\text{-}1}} + W_{\mathrm{cf}}c_{\mathrm{t\text{-}1}} + b_{\mathrm{f}}) \tag{2}$$

$$\mathbf{c_t} = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3}$$

$$\mathbf{o_t} = \sigma(W_{\mathrm{xo}}x_{\mathrm{t}} + W_{\mathrm{ho}}h_{\mathrm{t\text{-}1}} + W_{\mathrm{co}}c_{\mathrm{t}} + b_{\mathrm{o}}) \tag{4}$$

$$\mathbf{h_t} = o_{\mathrm{t}} tanh(c_{\mathrm{t}}) \tag{5}$$

where *i*, *f*, *c*, *o* correspond to the input gate, forget gate, cell state and output gate activations, σ denotes the sigmoid activation function and *tanh* – the hyperbolic tangent; x is the input, t is the time step, the W-values are the corresponding connection-matrices, and the b-values are the biases for the connections. The memory learns to selectively decide which pieces of information to keep and which to forget, thus making it possible to remember input from many steps ago and to discard input that is not relevant to the current state.

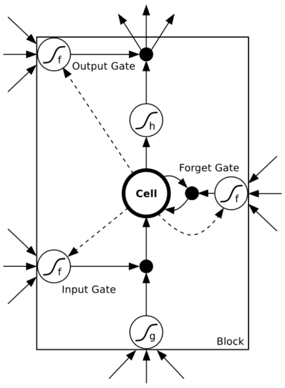


Fig. 3: Schematic representation of an LSTM block; figure taken from [7]

5 Shallow Neural Network Language Models

This brief overview concludes with a final note of potential interest. The description of the NN architectures in the previous sections featured a so-called *projection layer* in which input words (or one-hot vector representations of words) are translated into a lower-dimensional space where meaning is represented in a distributed manner. In the LMs described above the projection layer comes before the nonlinear transformation that happens in the hidden layer of the networks. The projection of the words can be learned together with the rest of the weights for the network, but at some point researchers found out that a two-step training process might be easier and even more effective (see for example [8]). At the first step of the process the word vectors are learned independently and at the second one they are directly plugged in the LM.

This insight led to a very important line of research that has enabled a qualitative leap in NLP in recent years. Previous to that, distributed representations of words (word embeddings) had been difficult to obtain, because the presence of the hidden layer increases drastically the computational complexity of training the models and makes the task prohibitively time-consuming when using large amounts of data.

[9] took this idea and applied it to training predictive models with huge amounts of data, but using much simpler architectures that decrease the training time from weeks to days and even hours, on corpora with billions of words. The key to this increase in speed is removing the hidden layer and using only a projection layer.

The same publication proposed two architectures: *c-bow* and *skip-gram*. The two follow essentially identical principles, but whereas the *c-bow* model tries to predict a single word surrounded by a context of N words, the *skip-gram* model does the reverse – predicting a surrounding context of N words based on the input of a word in its center. The two architectures offer different advantages, depending on the training data (e.g. usually *skip-gram* performs better when large amounts of data are available). Removing the hidden layer makes the output less precise, as can be expected, but also makes it possible to train word embeddings of useful sizes (e.g. 500-position vectors) on corpora that are truly representative of language in its infinite variations (e.g. the pre-trained vectors distributed by Google[1] are obtained after training on a 100-billion-words corpus). Further improvements of these models, as well as other approaches to embedding (see [10]) and the embedding of elements other than words (characters, suffixes), have been proposed since.

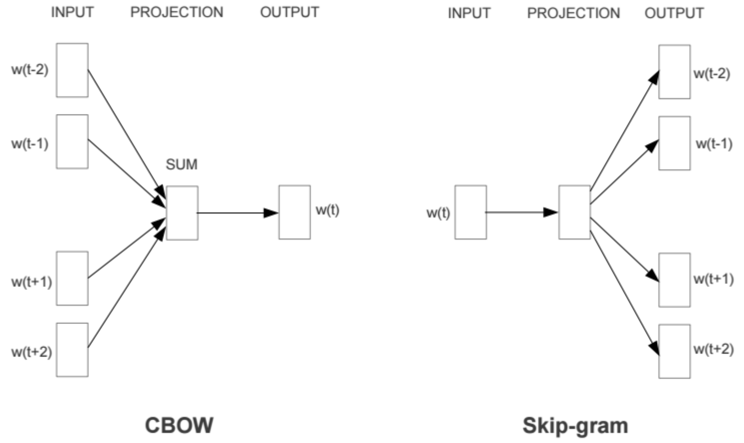


Fig. 4: The c-bow and skip-gram architectures; figure taken from [9].

The efficient calculation of distributed representations of words has enabled a lot of important research. The fact that this line of work is grounded in language modeling shows how powerful NNLM are for capturing dependencies in text. This is clearly illustrated by the astounding effects of performing simple algebraic operations on distributed representations, such as the example from [9] where *vector(”King”) - vector(”Man”) + vector(”Woman”)* produces a result that is very close to *vector("Queen")*. The multiple degrees of similarity captured by word vectors (syntactic, lexical, stylistic, etc.) show that something as simple as a LM can implicitly encapsulate a great deal of linguistic knowledge that may have been otherwise previously modeled in much more complex terms.

6 Conclusion

This article has presented an outline of the basic approaches to using neural networks for language modeling. It has compared the newer approaches to classic

[1] https://code.google.com/archive/p/word2vec/

count-based language models and has given brief explanations on how each of the problems associated with count-based LMs has been overcome using neural networks. Finally, the article discussed briefly the connection between language modeling and word vector representations which have become tremendously popular in the NLP community and have advanced significantly research in almost all of its areas.

Acknowledgements

This research has received partial funding from the EC's FP7 under grant agreement number 610516: "QTLeap: Quality Translation by Deep Language Engineering Approaches".

References

1. Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., and Trancoso, I.: Finding function in form: Compositional character models for open vocabulary word representation. In: arXiv preprint arXiv:1508.02096 (2015).
2. Popov, A.: Deep Learning Architecture for Part-of-Speech Tagging with Word and Suffix Embeddings. In: International Conference on Artificial Intelligence: Methodology, Systems, and Applications, Springer International Publishing, pp. 68-77 (2016).
3. Mikolov, T., Karafiát, M., Burget, L., Cernocký, J. and Khudanpur, S: Recurrent neural network based language model. In: Interspeech, Vol. 2, p. 3 (2010).
4. Sundermeyer, M., Hermann N., and Schlüter, R.: From feedforward to recurrent LSTM neural networks for language modeling. In: IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 23.3, pp. 517-529 (2015).
5. Cho, K., van Merrienboer, B., Bahdanau, D. and Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. In: arXiv preprint arXiv:1409.1259 (2014).
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. In: Neural computation 9.8, pp. 1735–1780 (1997).
7. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: Speech recognition with deep recurrent neural networks." Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, pp.6645–6649 (2013).
8. Mikolov, T., Kopecky, J., Burget, L., Glembek, O. and Cernocky, J.: Neural network based language models for higly inflective languages. In: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4725-4728 (2009).
9. Mikolov, T., Chen, K., Corrado, G. and Dean, J.: Efficient estimation of word representations in vector space. In: arXiv preprint arXiv:1301.3781 (2013).
10. Pennington, J., Richard S. and Manning, C.: Glove: Global Vectors for Word Representation. In: EMNLP. Vol. 14., pp. 1532-43 (2014).

Transfer of Deep Linguistic Knowledge in a Hybrid Machine Translation System

Kiril Simov, Alexander Popov, Lyubomir Zlatkov, and Nikolay Kotuzov

Institute of Information and Communication Technology, BAS,
Akad. G. Bonchev. 25A, 1113 Sofia, Bulgaria
{kivs,alex.popov,lyubo,nikolay.kotuzov}@bultreebank.org

Abstract. In this paper we present a hybrid architecture for machine translation using deep semantic transfer. The deep semantics is based on Robust Minimal Recursion Semantics. The hybrid architecture exploits a statistical machine translation model and transfer of linguistic information from source to target language used for post processing.

Keywords: Minimal Recursion Semantics, Deep Semantic Transfer

1 Introduction

The paper presents work done within the QTLeap project on deep semantic transfer in Machine Translation (MT). Under deep semantics in this paper we understand the *Logical Form* (LF) of a sentence. From the different approaches to representing the LF we have selected Minimal Recursion Semantics (MRS) ([1]). MRS is an underspecified semantic representation in which the scope of the quantifiers is left open when there is not enough information to determine it. MRS is initially developed to support MT. Here we report on the implementation of a hybrid architecture for deep semantic transfer. It uses statistical MT (SMT) for the initial translation; another component then transfers linguistic information from the analysis of the source text to the target text and this information is used for post-processing of the output of the statistical MT ([2]). The linguistic information is transferred from the source to the target text via word alignment constructed during the SMT. The transferred information is only partial, but still useful for the post-processing task.

The structure of the paper is as follows: in Section 2 a short introduction to MRS is given, as well as a set-up for MT using MRS structures; Section 3 describes the hybrid architecture; Section 4 reports on the implementation of the hybrid system and its results; the last section concludes the paper.

2 Minimal Recursion Semantics and MT

Here we use a variation of MRS called Robust Minimal Recursion Semantics (RMRS). RMRS is introduced as a modification of MRS which captures the semantics resulting from a shallow analysis — see [3] and [4]. The main motivation for this development is the observation that currently no single system can do everything. Thus, [3, 4] proposes a semantic representation which allows to build a uniform semantic representation for both deep and shallow processing. RMRS separates the arguments from the predicates. Each predicate is represented via its name and its main argument, which depends on the part of speech — *referential index* for nouns and some pronouns or *event index* in other cases. In this way it is possible that the predicates and their arguments are added to the structure separately from each other. An RMRS structure is a quadruple

$$< hook, EPbag, argumentset, handleconstraints >$$

where a hook consists of three elements $l : a : i$, l is a label, a is an anchor and i is an index. Each elementary predication (members of $EPbag$) is additionally marked with an anchor[1] — $l : a : r(i)$, where l is a label, a is an anchor and $r(i)$ is a relation with one argument of appropriate kind — referential index or event index. The argument set contains argument statements of the following kind $a : ARG(x)$, where a is an anchor which determines for which relation the argument is defined, ARG is the name of the argument, and x is an index or a hole variable or handle (h) for scopal predicates. The handle constraints are not discussed here. The information in RMRS structures is represented with the help of unary and binary relations. The unary relations represent the grammatical features, elementary predicates, lemmas, the main arguments of the elementary predicates. The binary relations relate the anchors of the arguments of the elementary predicates and the anchor of the elementary predicate.

The MRS rule-based MT uses rewriting rules of the following format:

$$[\mathcal{C} :]\mathcal{I}[!\mathcal{F}] \to \mathcal{O}$$

where $\mathcal{I}$ is the *input* of the rule, $\mathcal{O}$ is the *output*. $\mathcal{C}$ determines the *context* and $\mathcal{F}$ is the *filter*[2] of the rule. $\mathcal{C}$ selects the positive and $\mathcal{F}$ the negative context for the application of a rule. For more details see [5]. This type of rules allow an extremely flexible transfer of MRS structures between the source and the target languages. After the application of the rules, the result is a complete MRS structure for the target sentence. The actual sentence is constructed with a deep generation grammar. The translation can be represented as follows:

$$Sent_S \xrightarrow{DeepGA} MRS_S \xrightarrow{DeepT} MRS_T \xrightarrow{DeepGG} Sent_T$$

where $Sent_S$ is the source sentence, $DeepGA$ is deep grammar for the analysis of the source language, MRS_S is the resulting MRS structure, $DeepT$ is the set of rules for the deep transfer, MRS_T is the target MRS structure, $DeepGG$ is a deep grammar for generation for the target language, and $Sent_T$ is the generated target language sentence. The main problem is that this approach requires a good deep grammar for the source language which produces complete MRS structures,

[1] The anchors determine the tokens which generate the corresponding elementary predicates and related arguments.

[2] Each of $\mathcal{I}$, $\mathcal{O}$, $\mathcal{C}$, and $\mathcal{F}$ are fragments of the MRS structure.

then a complete set of rules for transferring of the source language MRSes to the target language MRSes, and a generation grammar for the target language. There are not many languages equipped with such grammars and rules.

3 A Hybrid MT Architecture for Deep Transfer

One observation is that the steps from the source sentence $Sent_S$ to the target sentence $Sent_T$ establish an alignment between $Sent_S$ and $Sent_T$ on the token level: $Sent_S \xrightarrow{Align_T} Sent_T$. Thus, if we have two sentences $Sent_S$ and $Sent_T$ that are translations of each other, their MRS structures MRS_S and MRS_T and a token level alignment $\xrightarrow{Align_T}$ between the sentences, we can use the alignment to generate an alignment between the two MRS structures. We exploit this observation to define a hybrid MT architecture for deep semantic transfer. The hybrid machine translation system consists of three main steps (Figure 1).

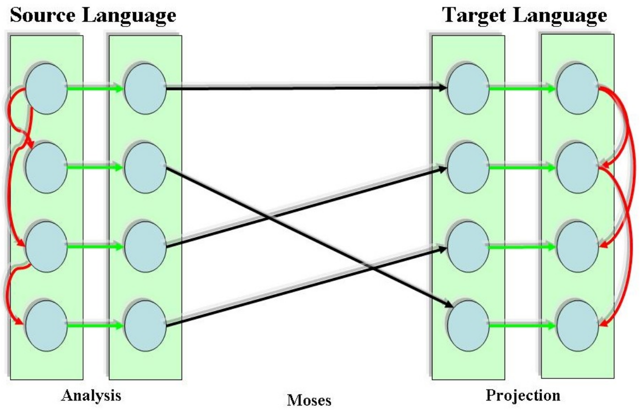


Fig. 1. A hybrid architecture for transferring linguistic information from the source to the target language. The linguistic analyses for the source language (Analysis - column 1) are projected to a tokenized source text (Analysis - column 2); then the Moses model (Moses) is applied to produce the target translation. The translation alignment (Projection - column 1) is used for transferring the information to the corresponding tokens in the target language (Projection - column 2). The projected linguistic information interacts with the linguistic features of the tokens in the target text. Finally, the resulting annotation of the target text is used for post-processing.

The source language analysis includes tokenization, POS tagging, Dependency parsing and RMRS structure construction over the result from the previous steps. Unfortunately the translated target sentence is often ungrammatical. Thus, we can provide only partial analyses of the sentence, keeping ambiguities in some cases. Here is an example of aligned texts annotated with morphosyntactic information – the English sentence "Place them in the midst of a pile of dirty, soccer kit." and its translation into Bulgarian[3]:

```
(place/VB them/PRP in/IN) = (postavyaneto/Ncnsd im/Ppetdp3;Ppetsp3;Pszt--3 v/R)
(the/DT midst/NN of/IN)   = (razgar/Ncmsi na/R)
(a/DT pile/NN of/IN)      = (kup/Ncmsi)
(dirty/JJ)                = (izmyrsyavam/Vpitf-r1s)
(,/,)                     = (,/Punct)
(soccer/NN)               = (futbolni/A-pi)
(kit/NN)                  = (komplekt/Ncmsi)
(./.)                     = (./Punct)
```

From the alignment and the rules for mappings between the two tagsets we could establish the following alignments on token level:

```
(them/PRP)   = (im/Ppetdp3;Ppetsp3;Pszt--3)
(in/IN)      = (v/R)
(midst/NN)   = (razgar/Ncmsi)
(of/IN)      = (na/R)
(pile/NN)    = (kup/Ncmsi)
(soccer/NN)  = (futbolni/A-pi)
(kit/NN)     = (komplekt/Ncmsi)
```

On the basis of these alignments, we were able to transfer additional information like dependency links, word senses and elementary predicates. It is clear from the example that the transfer is only partial. The alignment `(soccer/NN) = (futbolni/A-pi)` is allowed because this is a typical way to translate English Noun-Noun compounds into Bulgarian (see Table 1). After the transfer of linguistic information, a set of rules for post-processing are applied. For example, a rule for agreement between the adjective `futbolni` and the noun `komplekt` can be applied. In order to do this we perform the following processing steps.

Alignment Enrichment. First, we enrich the alignment generated during the SMT, making it more precise by creating token-to-token alignments where possible. For each sentence we have an alignment in the form of phrase alignments, where each phrase alignment is as follows: $(st_{i_1}\ st_{i_2}\ st_{i_3}\ \ldots\ st_{i_k}) \Leftrightarrow (tt_{j_1}\ tt_{j_2}\ tt_{j_3}\ \ldots\ tt_{j_m})$, where st_{i_n} is a source language token and tt_{j_o} is a target language token. The goal is to identify in the phrase alignments *token level alignments* of the following kind: $(st_{i_n}) \Leftrightarrow (tt_{j_o})$, assuming that the source language token st_{i_n} is translated into the target language token tt_{j_o}. In order to do this kind of alignment we exploit the source language morphosyntactic annotation and the potential target language annotation: $(st_{i_n}/stag_{i_n}) \Leftrightarrow (tt_{j_o}/ttaglist_{j_o})$, where $stag_{i_n}$ is the morphosyntactic annotation of the token st_{i_n} and $ttaglist_{j_o}$ is a list of the potential morphosyntactic tags for the token tt_{j_o}.

Such an alignment is constructed through an examination of all combinations of token correspondences from the phrase alignments. Then we formulate rules

[3] For English, the Pen treebank tagset is used: `https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html` and for Bulgarian, the BulTreeBank tagset is used: `http://www.bultreebank.org/TechRep/BTB-TR03.pdf`.

to expell the impossible ones from the list of candidate token alignments. The rules have the following form: $stagT \Rightarrow ttagT$, where $stagT$ is a template for source language morphosyntactic tags and $ttagT$ is template for target language morphosyntactic tags. For a candidate token level alignment $(st_{i_n}/stag_{i_n}) \Leftrightarrow (tt_{j_o}/ttaglist_{j_o})$ such a rule is applicable if the source language tag template matches the tag $stag_{i_n}$, then the target language tag template is evaluated with respect to each tag in the list $ttaglist_{j_o}$. If a target language tag cannot match the template, it is excluded from the list. If after application of the rule the list is empty, then the candidate token level alignment $(st_{i_n}/stag_{i_n}) \Leftrightarrow (tt_{j_o}/ttaglist_{j_o})$ is deleted from the candidate set. If the list is not empty, then we keep the resulting token alignment as a candidate: $(st_{i_n}/stag_{i_n}) \Leftrightarrow (tt_{j_o}/ttaglist^R_{j_o})$, where $ttaglist^R_{j_o}$ is the reduced list of tags. After the application of the rules, there are several possibilities of token alignment. We are interested in one-to-one mapping.

Linguistic Information Projection and Post-processing. The linguistic information to be projected is in the form of unary and binary relations over source tokens. The *unary relations* represent at least the following linguistic information: (1) elementary predicates and main arguments. The elementary predicates originate from tokens in the source text; (2) grammatical features. Grammatical features are transferred on the basis of a mapping between the tagsets of the source and the target languages. The *binary relations* correspond to relations between the main argument of an elementary predicate and one of its other arguments. The transferred information is used for application of post-processing rules. The whole procedure is as follows:

First, the transferred linguistic information forms a partial linguistic analysis of the target sentence.

Second, on the basis of the source sentence analysis and the (partial) target analysis we construct rules of the form discussed above: $[\mathcal{C} :]\mathcal{I}[!\mathcal{F}] \rightarrow \mathcal{O}$, but the rules here are more like templates in the sense that many elements in them are left underspecified. These rules determine the possible translations of the different types of RMRS structures from the source to the target language. In this way, the main argument type from the source language could be changed to a different type. When the rules are applied, each target token involved in the target RMRS needs to have at least one appropriate morphosyntactic tag in $ttaglist_{j_o}$ assigned to it that agrees with its elementary predicate type.

Third, the rules are instantiated with the corresponding tokens from the token level alignments. In this way, the template RMRS structures involved in the rules from step 2 become more concrete. This allows us to check some grammatical and syntactic characteristics of the tokens involved.

On the basis of the constructed target RMRS structure, we define post-processing rules in case certain conditions are not met by the RMRS structure. We consider here two actions: change of word form and change of word order. In the future we could extend the set of actions with insertion of tokens (clitics, subject, for instance), deletion of tokens, etc. The rules have the following form: $\langle tt_{i_n}, tt_{j_o}, r, cond\rangle \rightarrow action$, for a pair of target tokens $\langle tt_{i_n}, tt_{j_o}\rangle$ and relation r between them. If the condition $cond$ is not fulfilled, an action $action$ is performed.

The actions could be: $form(tt)$ — this action performs change of the word form for the token and it is used to ensure agreement, and $reorder(tt_{i_n}, tt_{j_o})$, used for reordering of the two tokens. Conditions are $agreement(tt_{i_n}, tt_{j_o})$ — it is true if the agreement conditions between grammatical features are met, $prec(tt_{i_n}, tt_{j_o})$ — it is true if the token tt_{i_n} precedes tt_{j_o}.

The rules could clash over the actions. For instance, there might be a case where two rules need to change the word order for two tokens. Here we do not consider this problem, assuming an order of rule application. Thus, only one action could be applied for a given case.

4 Implementation and Results

For the en→bg translation direction, the source language linguistic annotation consists of tokenization, POS tagging, lemmatization, dependency parsing, RMRS annotation. The analysis of English (tokenization, lemmatization, POS tagging and dependency parsing) as a source language was done with the CoreNLP tools[4] of Stanford University. The RMRS structures and the post-processing rules were implemented in the CLaRK System.[5] For the analysis of Bulgarian, we trained Mate tools[6] on the Bulgarian treebank. For the SMT translation and the construction of the phrase alignment we used a phrase-based Moses model as depicted in Figure 1. The alignment enrichment is done by rules in the CLaRK System. Once the linguistic annotation is projected, the post-processing rules can be applied.

type of change English→Bulgarian	frequency	example
noun1 noun2→noun1 noun2	rare	business meeting→biznes sresta
noun1 noun2→noun2 noun1	frequent	Rila mountain→planina Rila
noun1 noun2→adj1 noun2	frequent	antivirus software→antivirusni programi
noun1 noun2→noun2 prep noun1	frequent	email settings→nastrojki za posta

Table 1. Examples of structural changes in the translation of English noun-noun phrases into Bulgarian.

An example of a word order rule is the transformation of the English noun compounds into the appropriate syntactic structures in Bulgarian. The different templates are represented in Table 1. The direct transfer is rare, since the NN compounds are not so frequent in Bulgarian. The combination in which the first noun is a Named Entity is the most frequent one in the domain data. In the case of a phrase with an adjective and a noun in Bulgarian, a rule for word form

[4] `http://stanfordnlp.github.io/CoreNLP/`

[5] `http://www.bultreebank.org/clark/index.html`

[6] `http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/matetools.en.html`

change reflecting the agreement is applied. Agreement is checked in the case of verb elementary predicates and their subject arguments.

Language pair	Baseline	Deep Semantic Transfer
en→bg	20.30	23.91
bg→en	18.54	24.93

Table 2. BLEU scores for the translation between Bulgarian and English on the QTLeap Corpus Batch4. The Baseline system is a phrase-based Moses model.

By implementing these rules we have achieved the results presented in Table 2. The results in both translation directions show significant improvement.

5 Conclusion

The described hybrid architecture provides utilities to exploit deep semantic knowledge with relation to the transfer modules in machine translation. It is appropriate in cases when there is no good deep grammar for a given language or a domain. Our work on the projection of linguistic analyses from the source to the target text is similar to [6] and [7]. The main difference is that we project RMRS structures to the target language.

In the future we envisage using bilingual lexicons, parallel valency lexicons and word senses for better token level alignments and for the post-processing.

Acknowledgements

This research has received partial support by the EC's FP7 project: "QTLeap: Quality Translation by Deep Language Engineering Approaches" (610516).

References

1. Copestake, A., Flickinger, D., Pollard, C., Sag, I.: Minimal recursion semantics: An introduction. Research on Language & Computation **3**(4) (2005) 281–332
2. Simov, K., Osenova, P.: A hybrid approach for deep machine translation. In: Proceedings of the 2nd Deep Machine Translation Workshop. (2016) 21–28
3. Copestake, A.: Robust minimal recursion semantics (working paper). (2003)
4. Copestake, A.: Applying robust semantics. In: Proceedings of the 10th Conference of the Pacific Assocation for Computational Linguistics (PACLING). (2007) 1–12
5. Oepen, S.: The Transfer Formalism. General Purpose MRS Rewriting. Technical Report LOGON Project. Technical report, University of Oslo (2008)
6. Ramasamy, L., Mareček, D., Žabokrtský, Z.: Multilingual dependency parsing: Using machine translated texts instead of parallel corpora. The Prague Bulletin of Mathematical Linguistics **102** (2014) 93–104
7. Mareček, D., Rosa, R., Galuščáková, P., Bojar, O.: Two-step translation with grammatical post-processing. In: Proceedings of the Sixth WMT. (2011) 426–432

Towards Deeper MT: Parallel Treebanks, Entity Linking, and Linguistic Evaluation

Ankit Srivastava, Vivien Macketanz,
Aljoscha Burchardt, and Eleftherios Avramidis

German Research Center for Artificial Intelligence (DFKI Berlin),
Language Technology Lab, Alt-Moabit 91c, 10559 Berlin, Germany
firstName.lastName@dfki.de
http://www.dfki.de/web

Abstract. In this paper we investigate techniques to enrich Statistical Machine Translation (SMT) with automatic deep linguistic tools and evaluate with a deeper manual linguistic analysis. Using English–German IT-domain translation as a case-study, we exploit parallel treebanks for syntax-aware phrase extraction and interface with Linked Open Data (LOD) for extracting named entity translations in a post decoding framework. We conclude with linguistic phenomena-driven human evaluation of our forays into enhancing the syntactic and semantic constraints on a phrase-based SMT system.

Keywords: Machine Translation, Parallel Treebanks, Linked Open Data, Manual Evaluation

1 Introduction to Three Deep Language Processing Tools

Machine Translation (MT) like other language processing tasks is confronted with the Zipfian distribution of relevant phenomena. Although surface-data-driven systems have enlarged the head considerably over the last years, the tail still remains a challenge. Many approaches have therefore tried to include various forms of linguistic knowledge in order to systematically address chunks of the tail [1]. Unfortunately, todays automatic measures for MT quality are usually not able to detect these particular changes in the translations that may or may not constitute improvements. Therefore, we have argued for an evaluation approach that extends the current MT evaluation practice by steps where language experts inspect systems outputs [2]. We have started to use this extended evaluation approach in our contribution to the WMT2016 IT task [3]. In this paper, we will report more in-depth on three of the "deeper" ingredients of our work.

2 Baseline Machine Translation Systems

The experiment is based on two baseline systems: **Phrase-based SMT** follows several state-of-the-art phrase-based system settings as indicated in the Shared task of Machine Translation in WMT [4]. As the best system UEDIN-SYNTAX [5] included several components which were not openly available, we proceeded with adopting several settings from the next best system UEDIN [6]. In our system we follow the practice of augmenting the generic training data (Europarl [7], News Commentary, MultiUN [8], Commoncrawl [9]) with domain-specific data (Libreoffice, Ubuntu, Chromium [10]), and building relevant extensive language models, interpolated on in-domain data, as described above. **Rule-based MT** (RBMT) as in the transfer-based system Lucy [11] is also part of our experiment as a baseline, due to its state-of-the-art performance in many shared tasks. In this method, translation occurs in three phases, namely analysis, transfer, and generation. All three phases consist of hand-written linguistic rules.

The set of parallel sentences for training, and the development and test sets for tuning and testing respectively were sourced from the data provided for the WMT 2016 shared task on machine translation of IT domain [12], available at `http://www.statmt.org/wmt16/it-translation-task.html`.

3 Syntax-aware Phrase Extraction

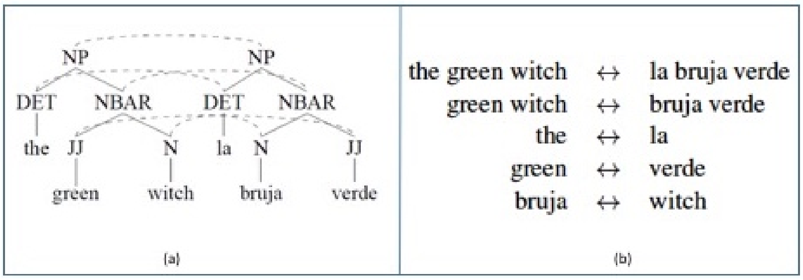
In this section we describe a syntax-aware enhancement to the phrase-based SMT baseline system described in Section 2. We extract linguistically motivated phrase pairs by obtaining phrase structure parse trees for both the source and target languages using monolingual constituency structure parsers such as the Berkeley Parser [13], and then aligning the subtrees using a statistical tree aligner [14]. These phrase pairs (illustrated with an example in Figure 1) are then merged with the phrase pairs extracted in the baseline SMT system into one translation model. Thus we are merely using syntax to constrain the phrase boundaries and enabling SMT decoder to pick syntax-aware phrases, thereby ensuring noun phrases and verb phrases remain cohesive. Through experimentation detailed in [15], we have discovered that non-linguistic (Base) phrase-based models have a long tail (of coverage) and syntax-aware phrases underperform, if not concatenated with non-linguistic phrase pairs. We observed the syntax-aware system scored 0.8 BLEU points over the baseline system.

Example (1) illustrates how the syntax-aware system (Syn) improves over the baseline SMT system (Base) by outputting the missing modal verb *ändern*.

(1) **Src (en):** You can **change** the screen saver settings.
Base (de): Sie können die Bildschirmschoner Einstellungen.
Syn (de): Sie können die Bildschirmschoner Einstellungen **ändern**.
Ref (de): Sie können die Bildschirmschoner-Einstellungen **ändern**.

4 Named Entity Translation Using Linked Data

Given the fact that our SMT system was not trained on data from the same domain as our testset (IT-domain), a number of technical terms (named entities) were either mistranslated or not translated consistently. One technique to



the green witch	↔	la bruja verde
green witch	↔	bruja verde
the	↔	la
green	↔	verde
bruja	↔	witch

Fig. 1. Example of (a) A parallel treebank entry and (b) The associated set of extracted phrase pairs.

address this is to integrate the SMT system with a Named Entity Recognition (NER) system. In this section, we describe our approach.

We exploit multilingual terms semantically linked with each other in the form of freely available linguistic linked data on the web such as DBpedia[1] to identify named entities in our dataset in the same vein of [16]. These entities and their linked translations are then forced upon the SMT decoder such that the Moses decoder favours these translations over those from the translation model. A step-by-step procedure is detailed in [17].

Example (2) illustrates how the term *MS Paint* is wrongly identified as a person in the baseline system (Base). On the other hand, the linked data system (Link) correctly disambiguates the entity.

(2) **Src (en):** **MS Paint** is a good option.
Base (de): **Frau Farbe** ist eine gute wahl.
Link (de): **Microsoft Paint** ist eine gute wahl.
Ref (de): **MS Paint** ist eine gute Möglichkeit.

5 Deep Manual Evaluation

We carried out an extensive manual evaluation of the performance of our MT systems described above. To this end, we created a domain-specific test suite with the objective of validating the systems' capabilities of specific linguistic phenomena.[2] Our method consists of the following steps: A linguist identifies systematically occurring errors related to linguistic phenomena in the output of the systems. 100 segments containing the respective phenomenon are randomly extracted for each linguistic category. The total occurrences of the phenomena in the source segments are counted in the selected sets and analogous in the

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

[2] We understand the term "linguistic phenomena" in a pragmatic sense, covering a wide range of issues that can impact translation quality.

system outputs[3]. The instances of the latter are divided by the instances of the former, giving the percentage of correctly translated phenomena.

The following example depicts the counting of instances of one of the linguistic phenomena, namely the menu item separators “>”. All systems except for the RBMT system correctly transfer the “>”.

(3) source: Go to Settings $\underline{>}$ iCloud $\underline{>}$ Keys. *2 inst.*
SMT: Gehen Sie zu Einstellungen $\underline{>}$ icloud $\underline{>}$ Schlüssel. *2 inst.*
RBMT: Gehen Sie zu Einstellungs$\underline{->}$ iCloud $\underline{>-}$Tasten. *0 inst.*
Syntax: Gehen Sie zu Einstellungen $\underline{>}$ icloud $\underline{>}$ Schlüssel. *2 inst.*
linked d.: Gehen Sie zu Einstellungen $\underline{>}$ icloud $\underline{>}$ Schlüssel. *2 inst.*

The linguistic categories that we found to be prone to translation errors in this context can be found in Table 1. For these categories, 2105 instances in 657[4] segments were found altogether. The overall average performance of the four systems at hand is rather similar, ranging from 71% to 77%.

Even though the **SMT** and the **RBMT** system have very similar overall average scores that outperform the other two systems, their scores on the phenomena are quite complimentary. The two linguistic extensions did not have strong effects on the performance of the systems on the error categories that we found error-prone (in pilot studies) and important for the given IT helpdesk domain. The **SMT-syntax** and the **linked data** system have similar overall scores and similar scores on the linguistic categories. What is particularly noteworthy is the only (negative) outlier, namely phrasal verbs. Precisely in this class, we would have hoped to see an improvement in performance of SMT-syntax. We will further investigate the reasons for this failure of dealing with phrasal verbs.

Table 1. Translation accuracy on manually evaluated sentences focusing on particular phenomena. Boldface indicates best systems on each phenomenon (row) with a 0.95 confidence level.

	#	SMT	RBMT	Syntax	linked d.
imperatives	247	68%	**79%**	68%	68%
compounds	219	55%	**87%**	55%	56%
“>” separators	148	**99%**	39%	**97%**	**97%**
quotation marks	431	**97%**	94%	93%	94%
verbs	505	85%	**93%**	81%	85%
phrasal verbs	90	22%	**68%**	7%	12%
terminology	465	**64%**	50%	53%	52%
average		**76%**	**77%**	71%	72%

[3] A detailed description of how to count the occurrences of the phenomena including explicit examples will be published elsewhere.

[4] For the phrasal verbs, only 57 instead of 100 segments could be extracted as this is a rather rarely occurring phenomenon.

6 Conclusion

We have described several ways of making machine translation more linguistically aware. We have attempted to introduce linguistically aware phrases in the models as well as show improvements in the translation of named entities by linking with semantic web resources such as the DBpedia. Our detailed evaluation of relevant linguistic phenomena has shown that the performance of several MT systems differs as do several ways of system combinations. Given this detailed method and results, it is now possible to select/improve systems with respect to a given task. The deep linguistic evaluation we have shown is task-based. In other domains and settings, other issues would need to be inspected. While reference-based automatic evaluation treats requirements of the task only very indirectly and measures "improvement on average", this direct, source-driven evaluation makes it possible to evaluate the performance and measure improvement on task-specific aspects. One obvious way for improving statistical systems would be to create targeted training material focussing on the relevant aspects such as imperatives starting from the test items.

7 Acknowledgment

This article has received support from the EC's Horizon 2020 research and innovation programme under grant agreements no. 645452 (QT21), from FP7 (2007-2013) under grant agreement number 610516: "QTLeap: Quality Translation by Deep Language Engineering Approaches", and by Project Digitale Kuratierungstechnologien (DKT), supported by the German Federal Ministry of Education and Research (BMBF), "Unternehmen Region", instrument "Wachstumskern-Potenzial" (No. 03WKP45). We also thank the two anonymous reviewers for their valuable comments.

References

1. Steedman, M.: Romantics and Revolutionaries: What Theoretical and Computational Linguistics need to know about each other. In: Linguistic Issues in Language Technology: 6(11) (2011)
2. Burchardt, A., Harris, K., Rehm, G., Uszkoreit, H.: Towards a Systematic and Human-Informed Paradigm for High-Quality Machine Translation. In: Proceedings of the LREC 2016 Workshop Translation Evaluation: From Fragmented Tools and Data Sets to an Integrated Ecosystem, Portoro, Slovenia (2016)
3. Avramidis, E., Burchardt, A., Macketanz, V., Srivastava, A.: DFKI's system for WMT16 IT-domain task, including analysis of systematic errors. In: Proceedings of the First Conference on Machine Translation, Berlin, Germany (2016)
4. Bojar, O., Buck, C, Callison-Burch, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Radu, S., Specia, L.: Findings of the 2013 Workshop on Statistical Machine Translation. In: Proceedings of the Eighth Workshop on Statistical Machine Translation, pp. 1–44. Association for Computational Linguistics, Sofia, Bulgaria (2013)

5. Nadejde, M., Williams, P., Koehn, P.: Edinburghs Syntax-Based Machine Translation Systems. In: Proceedings of the Eighth Workshop on Statistical Machine Translation, pp. 170–176. Association for Computational Linguistics, Sofia, Bulgaria (2013)
6. Durrani, D., Haddow, B., Heafield, K., Koehn, P.: Edinburghs Machine Translation Systems for European Language Pairs. In: Proceedings of the Eighth Workshop on Statistical Machine Translation, pp. 114–121. Association for Computational Linguistics, Sofia, Bulgaria (2013)
7. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. In: Proceedings of the tenth Machine Translation Summit, Volume 5, pp. 7986. Phuket, Thailand (2005)
8. Eisele, A., Chen, Y.: MultiUN: A Multilingual Corpus from United Nation Documents. In: Proceedings of the Seventh Conference on International Language Resources and Evaluation, pp. 2868–2872. European Language Resources Association (ELRA), La Valletta, Malta (2010)
9. Buck, C., Heafield, K., van Ooyen, B.: N-gram Counts and Language Models from the Common Crawl. In: Proceedings of the Language Resources and Evaluation Conference, Reykjavk, Iceland (2014)
10. Tiedemann, J.: News from OPUS A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In: N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, Advances in Natural Language Processing, volume V, chapter V, pp. 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria (2009)
11. Alonso, J.A., Thurmair, G.: The comprendium translator system. In: Proceedings of the Ninth Machine Translation Summit. International Association for Machine Translation (IAMT) (2003)
12. Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A.J., Koehn, P., Logacheva, V, Monz, C., Negri, M., Neveol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, L., Turchi, M., Verspoor, K., Zampieri, M.: Findings of the 2016 Conference on Machine Translation. In: Proceedings of the First Conference on Machine Translation at ACL 2016, pp. 131–198. Association for Computational Linguistics, Berlin, Germany (2016)
13. Petrov, S., Klein, D.: Improved Inference for Unlexicalized Parsing. In: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, New York (2007)
14. Zhechev, V.: Unsupervised Generation of Parallel Treebank through Sub-Tree Alignment. In: Prague Bulletin of Mathematical Linguistics, Special Issue on Open Source Tools for MT, Volume 91 , pp. 89–98 (2009)
15. Srivastava, A.: Phrase Extraction and Rescoring in Statistical Machine Translation. PhD Thesis, Dublin City University (2014)
16. McCrae, J.P., Cimiano, P.: Mining Translations from the Web of Open Linked Data. In: Proceedings of the Joint Workshop on NLP, LOD, and SWAIE, pp. 8–11. Hissar, Bulgaria (2013)
17. Srivastava, A., Sasaki, F., Bourgonje, P., Schneider, J.M., Nehring, J., Rehm, G.: How to Configure Statistical Machine Translation for Linked Open Data. In: Proceedings of the 38th Annual Conference on Translating and Computer, London, United Kingdom (2016)

Machine Translation for Crosslingual Annotation Transfer

Laura Tolosi[1], Valentin Zhikov[1], Andrej Tagarev[1], Kiril Simov[2], Petya Osenova[2], Gertjan van Noord[3], and Dieke Oele[3]

[1]Ontotext AD,
[2]Institute of Information and Communication Technologies, BAS,
[3]University of Groningen,
{laura.tolosi,valentin.zhikov}@ontotext.com;{kivs,petya,}@bultreebank.org;{g.j.m.van.noord,d.oele}@rug.nl

Abstract. In the paper two experiments for crosslingual annotation transfer via machine translation are introduced: named entity annotation transfer and sentiment annotation transfer. Both experiments show promising results for these tasks.

Keywords: Machine Translation, Crosslingual Annotation Transfer

1 Introduction

In this paper we present two experiments performed within the QTLeap project[1] on using machine translation to support the cost-effective production of annotated datasets for languages missing them. Our goal is to demonstrate that such datasets could support the language technology applications of interest. There are various cross-lingual tasks, whose completion might depend on MT. Such tasks are, among others, cross-lingual polarity detection [1], multilingual subjectivity analysis [2], named entity recognition [3], multilingual geoparsing [4]. Our work is similar as approach to [3] and [2]. The difference is the language pairs and also, for the named entities (NE) task – we translate all the corpus from one language to the other and then transfer the annotations back to the original corpus; for the sentiment analysis – we also generate a corpus in another language and transfer the annotations. The difference is that we use MaxEntropy classifiers instead of Naive Bayes and support vector machines (SVM).

The structure of the paper is as follows: the next section discusses the task, experimental set-up and results for the named entity annotation transfer. Section 3 outlines the task, experimental set-up and results for the sentiment annotation transfer. Section 4 concludes the paper.

2 Named Entity Annotation Transfer

The named entity recognizer is a central and critical processing component in the NLP technology and thus needs to be developed for each new language or

[1] http://qtleap.eu/

domain. In order to train and evaluate a named entity recognizer, an extensive specific dataset is needed, where the named entities are appropriately annotated. Given the volume of the dataset required, its manual annotation is time consuming and faces considerable costs, thus being prohibitive from a commercial point of view. Hence, alternative methodology needs to be sought, that is time-wise affordable and cost effective. In this experiment, machine translation is resorted to and experimented with as part of a fast and cheap procedure to develop appropriate language resources (and associated processing tools) for languages for which there may be no annotated datasets of the relevant kind.

In our experimental set-up the Named Entity Annotation Transfer was instantiated to the case of Dutch, and thus involving machine translation between this language and English. The steps are as follows:

1. A Dutch corpus annotated with Named Entities and linked to Dutch Wikipedia at Ontotext AD was translated into English with the QTLeap system;
2. The English outcome was annotated with Ontotext tools that support named entity recognition and identification in English. The result was processed with named entity annotation (classes: person, location, organization and a DBpedia identifier where available) for each sentence in the translated text;
3. The annotation from step 2, in turn, was transferred back to Dutch on the sentence base;
4. The precision and recall were calculated on the basis of the two annotations - the gold one and the transferred one;
5. The original Dutch corpus was annotated automatically using similar NE recognition and identification tools as for the English. Its performance was compared to the performance for the translated corpus.

The translation from Dutch to English was performed with an early version of QTLeap Pilot 3 system of Dutch to English QTLeap system. The annotation of the translated corpus was done with an existing pipeline for English Named Entity annotation. The annotations created in this way were compared to the gold standard annotation in the Dutch corpus. Also the original Dutch corpus was processed with the existing pipeline for Dutch Named Entity annotation. The two pipelines (for English and for Dutch) are comparable with respect to the types of the Named Entity they recognize. The main difference is in the corpora on which the two pipelines were trained and the version of the DBPedia used as a source of names (English or Dutch versions). For Dutch there was a domain corpus in addition to the general one. The results are presented in the Table 1.

Corpus	Precision	Recall	F-measure
The Original Dutch Corpus	0.86	0.84	0.85
Translated English Corpus	0.58	0.73	0.65

Table 1. Comparison of the two pipelines.

3 Sentiment Annotation Transfer

In this section we present the experiment and the results on transfer sentiment annotation from English to Bulgarian. The steps performed are as follows:

1. Selection of English corpus with sentiment (opinion) annotation;
2. Training classifier C_EN on the original corpus and calculation of F-measure;
3. Translation of the corpus to Bulgarian by QTLeap technology and transfer of the annotation on sentence base;
4. Training the same type of classifier C_BG on the translated corpus and calculation of F-measure;
5. Comparing both F-measures and finding out how much performance was ”lost by translation”.

Similarly, the translation from English to Bulgarian was performed with an early version of Pilot 3 system English to Bulgarian QTLeap system. We analyzed the customer review dataset that has been introduced in [5]. The dataset is publicly available at: https://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html. For the sentiment analysis the data was further processed in the following way. The dataset consists of product reviews, manually annotated. Each sentence that mentions some opinion about a feature of a product is annotated with feature [+n] . Example:

`looks[+2] battery[-3]: I like the looks of the phone but the battery life is too short.`

We aggregated the scores into one final label of the sentence, namely +2 -3 = -1.

In this way we evaluate sentences with negative, positive or null scores. The latter are the most numerous, since sentences not containing any opinion about any feature are also labeled as null. We obtained a three-class classification problem: negative, positive and neutral sentences. Each of those are available in English (from the original dataset), and in Bulgarian (after translation with QTLeap technology). We trained MaxEntropy classifiers both on the English and the Bulgarian corpus. Classifiers were evaluated based on their F1 score. The F1 score is estimated via a cross-validation procedure, performed after combining all data (from all five products), shuffling and splitting in 5 bins for 5-fold

Sentiment	Precision	Recall	F-measure
English			
Classifier	65.59	65.59	65.59
Baseline	36.03	56.34	43.96
Bulgarian			
Classifier	65.05	65.05	65.05
Baseline	35.65	55.41	43.39

Table 2. Comparison of the sentiment analysis modules.

cross validation. We ran the most basic bag-of-words model, without stemming, because we imagined that we have no language resources in the target language (Bulgarian in this case). The results are presented in Table 2.

4 Conclusion

Here we presented the results from two experiments using MT for crosslingual annotation transfer. For Task 1 the drop in the performance (especially Precision) is more substantial because of two reasons: (1) difference in Named Entity annotation pipeline for the two languages; (2) features used by the pipeline. The main difference of the pipelines are the two different versions of DBpedia used as sources for the classification of the named entities. Because the English DBpedia is much bigger than the Dutch one the classifier for English had to select among more alternatives than the classifier for Dutch. The features used by the pipelines are the same for the two pipelines and include the links inside DBPedia, the frequency of NE types as well as their context in the text.

For Task 2 the drop in the performance is small, because of the method used for the sentiment analysis - bag of words. In this case translation does not damage the classification substantially because the word order and agreement do not play an important role. A useful result from the experiments is the list of positive and negative features for Bulgarian. These features could be used to facilitate manual annotation of original reviews in Bulgarian.

Despite the limited resources invested in the experiments, the results demonstrate the utilities of QTLeap translation services. Exploitation of MT for annotation transfer is feasible and could be a good starting point for further development of appropriate annotated corpora in a semi-automatic way.

Acknowledgements

This research has received partial support by the EC's FP7 project: "QTLeap: Quality Translation by Deep Language Engineering Approaches" (610516).

References

1. Demirtas, E., Pechenizkiy, M.: Cross-lingual polarity detection with machine translation. In: Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining. WISDOM '13, ACM (2013) 9:1–9:8
2. Banea, C., Mihalcea, R., Wiebe, J., Hassan, S.: Multilingual subjectivity analysis using machine translation. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, October 2008. (2008) 127–135
3. Dandapat, S., Way, A.: Improved named entity recognition using machine translation-based cross-lingual information. In: Proceedings of CICLING 2016. (2016)
4. Chen, X., Zhang, H., Gelernter, J.: Multi-lingual geoparsing based on machine translation. In: http://www.cs.cmu.edu/ gelernter/pdfs/Multi-lingual.pdf. (2015)

5. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proc. of KDD '04, ACM (2004) 168–177