Integrating Encyclopedic Knowledge into Neural Language Models

Yang Zhang, Jan Niehues, Alexander Waibel

Institute for Anthropomatics
Karlsruhe Institute of Technology, Germany
firstname.lastname@kit.edu

Abstract

Neural models have recently shown big improvements in the performance of phrase-based machine translation. Recurrent language models, in particular, have been a great success due to their ability to model arbitrary long context. In this work, we integrate global semantic information extracted from large encyclopedic sources into neural network language models. We integrate semantic word classes extracted from Wikipedia and sentence level topic information into a recurrent neural network-based language model. The new resulting models exhibit great potential in alleviating data sparsity problems with the additional knowledge provided. This approach of integrating global information is not restricted to language modeling but can also be easily applied to any model that profits from context or further data resources, e.g. neural machine translation. Using this model has improved rescoring quality of a state-of-the-art phrase-based translation system by 0.84 BLEU points. We performed experiments on two language pairs.

1. Introduction

Recurrent neural network language models (RNNLMs) have recently shown great improvements in statistical machine translation (SMT), both during decoding and rescoring. The use of continuous word representations has achieved better generalization of the data which effectively lowered data sparsity problems. Furthermore, the recurrent connections are able to model long-range dependencies. Yet, most of these models strictly depend on monolingual and parallel data, which is sometimes not available in large amounts or only for specific domains, especially with low-resource languages. This has motivated RNNLMs to take multiple parallel streams of data as input instead of just the single stream of surface words. These so-called factors can be used to add additional information, e.g. part-of-speech (POS) or automatic word clusters, which helps mainly with morphologically rich languages (e.g. Romanian, German). However, so far the additional factors were only limited to syntactic or local context information preceding the current word. Especially for languages without sufficient training data it is important to take advantage of other knowledge sources, e.g. encyclopedia. For example, Wikipedia has become a growing source for learning general concepts since the emergence of the Internet has led to an explosion of textual data. In this paper, we studied the integration of large encyclopedia sources into RNNLMs by using two approaches. In the first approach we use a factored model to integrate Wikipedia categories.

In order to understand large unstructured data sets great achievements have been made in latent concept learning in the area of information retrieval. Techniques that categorize documents include latent semantic analysis and probabilistic topic modeling. In the second approach, we use term frequency–inverse document frequency (tf-idf), latent semantic analysis (LSA) and latent Dirichlet allocation (LDA) to compute a real-valued topic vector for each sentence that is fed as an additional input into the network.

These approaches utilize external word categories and topic information in addition to the local contexts implicitly provided by recurrent neural models to improve lexical selection.

2. Related work

Language models are a critical component of many application systems, e.g. automatic speech recognition, machine translation and optical character recognition. However, language models have always faced the problem of data sparsity. Factored language models [1] introduced the use of a bundle of factors associated with a word which outperformed previous n-gram models without expanding the training data. In the work of [1] morphs, stems, POS and clustered word classes were used as factors. In [2], the single feature stream of surface words was replaced with multiple factors and integrated into a phrase-based SMT system by breaking down the translation model into several steps that pertain to the translation of single factors, which again influence the prediction of the next word. After recurrent neural network models became a success in language modeling [3], a factored input layer was employed in a model by [4] which uses a structured output layer based on word classes to handle vocabularies of large size. Motivated by multi-task learning in natural language processing, [5] proposed a multi-factor recurrent neural network language model (FRNNLM) which jointly predicts different output factors by mapping the output of the LSTM-layer [6] to as many softmax layers as there are output factors, thereby creating multiple distributions at the output layer. In the rescoring of n-best lists, this model can

be included as either one or several additional features depending on whether the output is treated as a joint probability or individual probabilities. Compared to a fixed-length factored input, a dedicated continuous space vector offers more flexibility and possibility to integrate additional side information, e.g. topic information. In [7], an approach to use an additional input vector in a neural language model was proposed. In their work, a vector instead of a single factor is associated with each word. However, this vector depends only on a word's earlier local context, thus neglecting the influence of the future context on a word's meaning. Often, the meaning of a word cannot be just derived from its preceding words but depends on the content words of the entire sentence or surrounding sentences. Topic models play a great role in information retrieval because they summarize large amounts of documents into fewer concepts by capturing word co-occurrence information. Essentially, topic models can be divided into vector space models (VSMs), e.g. LSA [8], and probability models, e.g. LDA [9]. The successful usage of Wikipedia to devise methods for computing semantic relatedness of documents was reported in [10] and [11]. Generative probabilistic models were employed in [12] to link named entities in text documents by using information extracted from Wikipedia. In [13] and [14] VSMs were employed to resolve word disambiguations based on entities derived from Wikipedia. Similar methods were applied to other knowledge sources, such as WordNet [15]. However, to our knowledge, it is the first time encyclopedic knowledge is used to supplement neural language modeling.

3. Integration of encyclopedic information

We propose two approaches to integrate encyclopedic information into neural language models of the target side of a language pair. First, the target side words are labeled with Wikipedia categories. The hierarchical page structure of Wikipedia in the target language enables us to obtain a page's category which we use as factored input into the previously described FRNNLM. Second, for every sentence in the data a ranking of related encyclopedia articles is established by using a VSM or topic model. Based on the underlying model, a feature vector, which comprises the information of the most similar documents to the current sentence, is computed and fed into a neural language model as additional input. The second approach is not limited to Wikipedia, but can be applied to any encyclopedia. To test the performance of this method independently of the underlying knowledge source we have crawled a Chinese lexicon from zdic.net [16] with 38.285 articles, for which the results are presented later on.

3.1. Word-level information

The motivation behind using word categories from Wikipedia in association with words is to strengthen the relatedness of words in the same correct translation hypothesis to prevent it from being discarded due to a low translation score. Consider the following example of a source sentence:

"A journalist writes articles for a column"

Both of the following hypotheses, as shown in Example 1, are possible translation hypotheses:

Example 1: *Two hypotheses whose nouns are labeled with Wikipedia categories.*

1. 一位 (one) 记者 (journalist) 给 (for) 柱子 (column) 写 (write) 文章 (article)
   NR 新闻 (=news) P 建筑(=archit.) VV 作品
2. 一位 (one) 记者 (journalist) 给 (for) 专栏 (column) 写 (write) 文章 (article)
   NR 新闻 (=news) P 新闻(=news) VV 作品

Although the second translation is more accurate, because "column" in the context of "journalist" and "article" is translated with a newspaper section, the first translation could be more likely according to the translation model, since, generally, a column refers more often to a pillar than a newspaper section. In both translations, the translation of "column" is three words away from its next context word, the translation of "article". In a translation task where the data does not originate from news-related domains or is limited in size such that the words "专栏 (=column)" and "记者 (=journalist)" are never seen in the same context, the language model will fail to choose the correct translation. Our approach solves this problem by labeling the words with according Wikipedia categories. By doing so, the bond between "专栏 (=column)" and "记者 (=journalist)" would be reinforced by their common factor "新闻 (=news)", which contributes to a higher score of the correct translation.

3.1.1. Input representation

We are only interested in Wikipedia pages, that are either an article or a category page. We define the search space as the set of all Wikipedia pages of interest. Given a word, we search for the page with the same word as title and retrieve its category which is found at the bottom of the page. In case of multiple categories we pick the first one. In the implementation, the offline Wikipedia dump is used to create a mapping between pages and their categories. Using a lower bound for the number of elements in a category, we pick recursively the parent of a category if it does not meet this requirement. In the later experiments we used a lower bound of five. This reduces the total number of associated articles and leads to better performances. In case no category is found at all, a word's part-of-speech is used instead. One characteristic of Wikipedia is that most of its articles only refer to a small group of lexical categories. For example, there are a lot more articles about objects (nouns) than activities (verbs). Since topic words are usually nouns, it suffices to label only nouns

in a sentence to improve model quality, which is shown in Example 1.

3.1.2. Factored neural language model

To incorporate the Wikipedia factors, we trained FRNNLMs as proposed in [5], which takes one or multiple factors at the input layer and offers the option of a factorized output layer. After concatenating the embeddings of the input factors into a single word embedding, this vector is sent through one or multiple LSTM-based layers [6] before projected onto the factored output probabilities. The instance of the model used for this work takes two factors, the surface form and the Wikipedia word category. These are mapped onto an embedding vector of size 100, which is the same size as the first LSTM-layer. For the second LSTM-layer 200 nodes are chosen. We defined only one output factor, which is the word's surface form.

3.2. Sentence-level information

The idea to use a dedicated topic vector is motivated by a more flexible representation of information and methods that are capable of drawing correlations betweens words independently of their distance. For example, the sentence

"Columns contain articles written by journalists"

exhibits several problems. First, when a corpus is translated into a low-resource language, e.g. Romanian, some of the content words do not have their own Wikipedia entries. This means we cannot take full advantage of the first approach. Second, the main context word of "columns", which is "journalists", appears later in the sentence; therefore a recurrent language model that only considers the previous context would fail. This motivated the idea to consider words in the entire sentence, unlike [7], and create a topic vector for each sentence by searching through Wikipedia to find topic related articles. In the above example sentence, these would probably be news-related articles. By compressing the information of these articles into a vector which is associated with each word, external topic information spanning neighboring words can be considered when translating a word. To create a ranking of similar documents and their representation to a given sentence, VSMs can be employed, such as tf-idf [17] and LSA [18], but also probabilistic models, such as LDA [9]. After representing the sentences and cleaned encyclopedia documents $\mathrm{D} = \{d_1, d_2, \ldots, d_M\}$ in vector form, where the vector entries index into a vocabulary $V = \{v_1, v_2, \ldots, v_N\}$, we query for each sentence $w = \{w_1, w_2, \ldots, w_l\}$ the n most similar documents $\mathrm{D}_n = \{\ d \mid \mathrm{score}(w, d) > c\ \}$, where $c \in \mathbb{R}$ is a threshold such that $|\mathrm{D}_n| = n$ and $\mathrm{score}(w, d)$ is a similarity function characterized by chosen model. These documents provide topic-related information about the current sentence. D_n are transformed into vector representations $\{\ h(d) \mid d \in \mathrm{D}_n\ \}$, where the vector representation $h(d)$ is defined by the model,

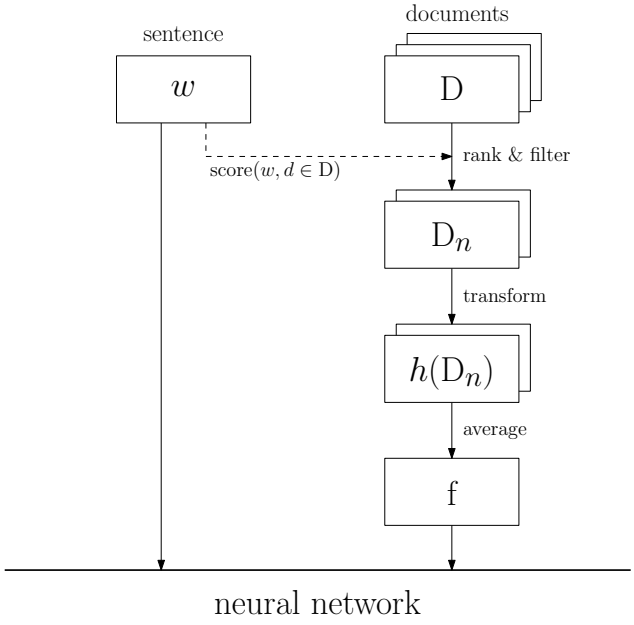

Figure 1: *Feature input creation process for the ERNNLM. Given a sentence w, encyclopedia articles are scored based on their topic relatedness and represented in vector form. The average of vectors of the most similar articles constitutes the feature input* f.

which can differ from that used for scoring. The average of the vectors forms an additional feature input f for a recurrent neural network-based language model, which we will refer to as an extended recurrent neural language model (ERNNLM).

$$\mathrm{f} = \frac{\sum_{d \in \mathrm{D}_n} h(d)}{|\mathrm{D}_n|} \tag{1}$$

This process is illustrated in Figure 1. For the topic models we used the python library gensim [19].

3.2.1. Tf-idf

Tf-idf [17] is a co-occurrence measure and determines the importance of a word to a set of documents D. Having the capability of grading down terms that appear in multiple documents, tf-idf is computed by multiplying a local component (term frequency or tf) with a global component (inverse document frequency or idf). The cosine distance can be used to determine a ranking of similar documents for a given sentence query. Shortcomings of this model include the inability to reduce the description length of the document, since words are only replaced by values. Given a query sentence w and a document d, we define the scoring function for this model as

$$\mathrm{score}_{tfidf}(w, d) = \frac{\sum_{i=1}^{N} tfidf(v_i, w, \mathrm{D}) \cdot tfidf(v_i, d, \mathrm{D})}{\| \sum_{i=1}^{N} tfidf(v_i, w, \mathrm{D}) \| \cdot \| \sum_{i=1}^{N} tfidf(v_i, d, \mathrm{D}) \|} \tag{2}$$

The vector representation of a document is defined as

$$h_{tfidf}(d) = (tfidf(v_i, d, D))_{v_i \in V} \quad (3)$$

3.2.2. Latent semantic analysis

LSA [8] is a method that discovers hidden concepts in documents by using single value decomposition (SVD) on the set of documents D. LSA uses a term-document matrix A, where the entry $A_{i,j}$ equates to the tf-idf value of the term i for the document j. SVD decomposes A into U, Σ, and V, such that $A = U\Sigma V^T$. LSA uses the decomposition to find a low-rank approximation, that is, a matrix $A_k = U_k \Sigma_k V_k^T$ of a predefined lower rank k closest in similarity to the original matrix A. This is done by deleting all but the k biggest single values in Σ. By doing so, LSA minimizes the Frobenius norm $\|A - A_k\|_F$. In this work, k is the number of hidden concepts to be learned. The approximation of the i-th document vector in the low-dimensional space, denoted with $\hat{d}_i$, is the i-th column of V_k^T. The number of dimensions k is determined empirically. Essentially, k is much smaller than the original space dimension, which is usually the total number of documents. Previous papers show that for Wikipedia dumps a good value for k should be chosen between 200 to 500 [18]. In this work k is 300. Given a sentence w and a document d, we define the scoring function for this model as

$$score_{lsa}(w, d) = \frac{\Sigma_k^{-1} U_k^T w * \hat{d}}{\|\Sigma_k^{-1} U_k^T w\| \cdot \|\hat{d}\|} \quad (4)$$

The vector representation of a document is defined as

$$h_{lsa}(d) = \hat{d} \quad (5)$$

3.2.3. Latent Dirichlet allocation

LDA [9] is a generative probabilistic model that discovers automatically topics from a data collection. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. The model is a three-level hierarchical Bayesian model with the first level being the corpus-level, the second being the document-level, and the third being the word-level. Setting the number of topics to be learned to k, learning is performed with variational Bayesian inference in combination with the EM algorithm. As for k, [20] discusses how to choose the number of topics. In this work k is 100, which is also the size of the feature vector. Given a sentence w and a document d, we defined the scoring function for this model as

$$score_{lda}(w, d) = P(w|\theta_d, \alpha, \beta), \quad (6)$$

where θ, α and β are parameters denoted and estimated as in [20]. The vector representation of a document is defined as

$$h_{lda}(d) = \theta_d \quad (7)$$

3.2.4. Extended language model

The basic RNNLM consists of an input layer, a hidden layer with recurrent connections that maintains a representation of the sentence history, and an output layer which produces the probability distribution over words. The ERNNLM, illustrated in 2, extends the RNNLM with an additional sentence-based feature layer that is connected to the output layer. Since this real-valued feature vector stays the same for all words in the current sentence, it is replicated for each word. This way, the feature information is retained in the model while the same sentence is being processed. The m hidden layers are based on LSTMs [6]. Given a sentence $w = \{w_1, w_2, \ldots, w_l\}$, the sentence-level information f is computed according to (1) and duplicated for each word. For the i-th word we denote its 1-of-N representation with x_i, the hidden layers with $s^1, s^2, \ldots, s^m$ and the output layer with y_i. The hidden and output layers are computed according to (8).

$$\begin{aligned} s_i^1 &= f_1(U_1 x_i + W_1 s_{i-1}^1) \\ s_i^j &= f_j(U_j s_i^{j-1} + W_j s_{i-1}^j), \\ & \quad j \in \{2, \ldots, m\} \\ y_i &= g(V s_i^m + F\mathrm{f}) \end{aligned} \quad (8)$$

where f_i represents activation functions, and g the softmax function. To train the network, that is to find the weight matrices $U_{1,\ldots,m}, V, W_{1,\ldots,m}, F$, stochastic gradient descent is used according to the negative log-likelihood loss function. We also tested the option of adding an additional connection from the feature layer to the first hidden layer, which is illustrated in Figure 2.

4. Experiments

We evaluated both the FRNNLM and the ERNNLM on English-Chinese (ZH) and English-Romanian (RO) language pairs. For each language pair n-best lists are created with our in-house phrase-based MT system; the models are used as additional features in rescoring.

4.1. System description

The baseline system is an in-house implementation of a phrase-based MT system and is used to generate n-best lists on all of the available training data. The ZH-system is trained on the TED and UN corpus, optimized and rescored on the TED dev2010 and tested on the test2010 corpora. The Chinese corpus contains 148.968 sentences, the development set 887 sentences. The Chinese neural language models are tested on a 3000-best list with a total of 455.965 sentences. The RO-system is trained on the corpora of WMT 2015 Shared Translation Task, optimized on the first half of news-dev 2016 and tested on the second half of news-dev 2016. In addition, a subset of 2000 sentences of the SETimes corpus is used for further optimization in rescoring. The Romanian corpus contains 604.588 sentences, the development

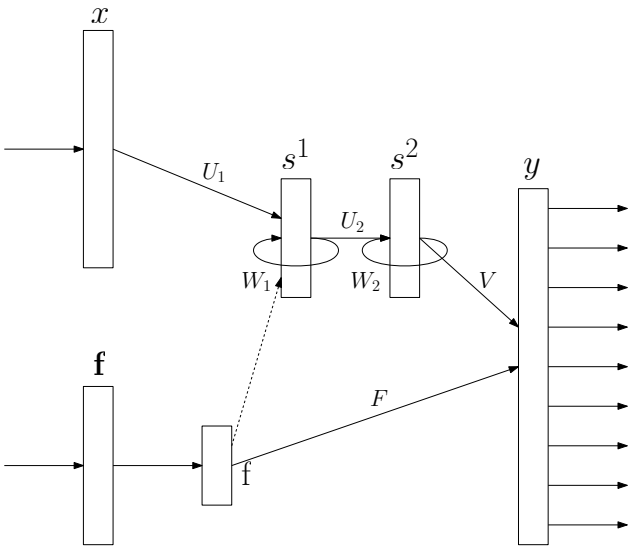

Figure 2: *ERNNLM with additional feature input* **f** *and two LSTM-based hidden layers. The dashed line from the feature input to the first hidden layer represents an optional connection.*

set 1000 sentences. The Romanian neural language models are tested in three configurations on 300-best lists with 1.288.319, 167.567 and 162.928 sentences respectively.

The ZH-baseline system uses two word-based language models, and 12 features in total to create a 3000-best list. The RO-baseline system uses two word-based language models, two cluster-based models using 50, 100 or 1000 clusters, and a POS-based language model. In total 22-23 features are used to generate a 300-best list. A full system description can be found in [5]. In decoding, both language pairs are optimized with minimum error rate training (MERT) [21]. The same is used for rescoring of the ZH-system, whereas the RO-system uses ListNet to rerank the n-best list. Both FRNNLMs and ERNNLMs for both language pairs are trained on the target side of the parallel training data. The Chinese neural language models use a vocabulary of 10K, while the Romanian models use a vocabulary of 5K. Wikipedia is used to obtain word-level side information, as described in Section 3.1. To gather sentence-level topic information for the ZH-system, as described in 3.2, we additionally web-crawled a Chinese lexicon from zdic.net [16] to show the model's performance independently of a specific encyclopedia.

Table 1: *ZH-FRNNLM*

Model	devdata	testdata
Baseline	14.7	17.02
+FRNNLM POS	14.77	16.97
+FRNNLM WikiCat	14.89 (+0.12)	17.63 (+0.66)
+FRNNLM WikiCat + POS	14.75 (-0.02)	17.81 (+0.84)

Table 2: *ZH-ERNNLM: overview of different feature vectors*

score(w,d)	h(d)	devdata	testdata
Baseline		14.70	17.02
TFIDF	TFIDF	14.78 (+0.08)	17.68 (+0.66)
LSA	TFIDF	14.78 (+0.08)	17.31 (+0.29)
LSA	LSA	14.83 (+0.13)	17.80 (+0.78)
LDA	TFIDF	14.79 (+0.09)	17.41 (+0.39)
LDA	LDA	14.79 (+0.09)	17.27 (+0.25)

4.2. English-Chinese

The ZH-system before rescoring gives a BLEU score 17.02 in testing. For all of the following experiments, our baseline system is rescored with a basic RNNLM. In the first ZH-experiment we used the scores of the FRNNLM in addition to the baseline features, which is shown in Table 1. Only nouns in the data are labeled with Wikipedia categories, which leads to 11.21 % coverage of the development set and 10.61 % of the test set. The FRNNLM that uses words and Wikipedia categories as factors causes an increase of 0.66 BLEU in testing compared to the same model without Wikipedia categories. The system that uses words, Wikipedia categories, and POS as factors performs 0.84 BLEU points better than the same system without the use of Wikipedia categories.

In a second experiment, the ERNNLM along with the different methods to compute the feature input vector based on Wikipedia are studied. Tf-idf, LSA and LDA are used for similar document ranking as well as vector representation. It is worth mentioning that the method for ranking can be paired with a different choice for representation. The performance results of various combinations of pairings are presented in Table 2. Although all combinations result in a better rescoring performance than the baseline system, choosing the same method for both steps attains a higher gain for the tf-idf and LSA model. For example, the tf-idf model creates an increase of 0.66 BLEU on the baseline system, and LSA an increase of 0.78 BLEU. The only exception to this rule is LDA. One reason for this could be suboptimal parameter picks for this generally more complex generative model. Despite the slightly better performance of the LSA model, the tf-idf model is employed in the following experiments due to its simplicity and, thus, faster training and evaluation.

We also tested the ERNNLM on a Chinese lexicon crawled (zdict) from zdic.net [16]. It contrasts Wikipedia

Table 3: *ZH-ERNNLM: comparison between different encyclopedia sources*

Model	devdata	testdata
Baseline	14.7	17.02
+ERNNLM WIKI	14.78 (+0.08)	17.68 (+0.66)
+ERNNLM ZDICT	14.91 (+0.21)	17.58 (+0.56)

Table 4: *ZH model perplexities*

Model	PPL
Baseline	128.17
FRNNLM POS	110.86
FRNNLM WikiCat	109.73
FRNNLM WikiCat+POS	110.38
ERNNLM WIKI	118.11
ERNNLM ZDICT	118.64
ERNNLM WIKI+ZDICT	119.02
ERNNLM WIKI 4-CONTEXT	118.46

in the variety and length of definitions. The lexicon provides short but precise Chinese explanations for all types of words, particularly verbs. Despite the differences between Wikipedia and zdict, the use of the lexicon achieves an increase of 0.56 BLEU in testing, which is comparable to the model improvement based on Wikipedia, as shown in Table 3.

An overview of all the models discussed so far along with their perplexities is given in Table 4. All models exhibit an evident reduction in perplexity compared to the baseline system, which is consistent with the rescoring results. In addition, two ERNNLMs whose feature inputs are determined differently are listed with their model perplexities. The first model's feature input depends on both Wikipedia and zdict; the second model's feature input takes a context of four past and future sentences into account.

As indicated in Figure 2, an additional connection was established between the feature layer and the first hidden layer in another experiment. As a result, the ERNNLM with two connections gained a small increase of 0.16 BLEU on the ERNNLM with just one connection and a total of 0.79 BLEU on the baseline system, as shown in Table 5.

Table 5: *ZH-ERNNLM: connecting feature input with two layers*

Model	devdata	testdata
Baseline	14.70	17.02
+ERNNLM TFIDF	14.78 (+0.08)	17.68 (+0.63)
+ERNNLM 2Conn TFIDF	14.74 (+0.04)	17.81 (+0.79)

Table 6: *RO-FRNNLM: word coverage of Wikipedia categories*

Data	Nouns	All
Devdata	1.94%	9.06%
Testdata	2.62%	10.17%
Setimes	3.48%	11.14%

Table 7: *RO-FRNNLM: single scores*

Input	Prediction	Single
Word	Word	27.88
4F	4F	28.54
+WikiCat (N)	+WikiCat (N)	28.71 (+0.17)
+WikiCat (A)	+WikiCat (A)	28.84 (+0.30)

4.3. English-Romanian

In the previous work [5], the FRNNLM for English-Romanian integrated four factors: the word's surface form, POS, and word clusters with 100 and 1000 classes respectively. Our Romanian neural language models build on top that, using a vocabulary size of 5K for all systems and the same notations to illustrate the following results. In the first experiment, where the results are shown in Table 7, the FRNNLMs are evaluated without the baseline features. The system from the previous work that uses all four factors (4F) for input and prediction has a BLEU score of 28.54. This model will serve as reference for our models. For the RO-system we chose to extract Wikipedia categories not only for nouns (N) but also for all word types (A). The word coverages of labeled words are displayed in Table 6, which shows significant differences between the two options. After adding the extracted Wikipedia word categories as additional factors, the FRNNLMs show an improvement of 0.17 BLEU and 0.30 BLEU respectively depending on how the data was labeled.

Table 8 shows the results of the model's joint probability used in addition to the baseline features in three configurations using different features [5]. Adding Wikipedia categories for all word types (A) achieves an improvement of about 0.1 BLEU in two configurations (Conf2 and Conf3) over the reference model.

Table 8: *RO-FRNNLM: end scores*

Model	Conf1	Conf2	Conf3
Baseline	29.86	30.00	29.75
+FRNNLM 4F	29.94	30.01	30.01
+FRNNLM 4F+N	29.94	30.31	29.99
	(+0.00)	(+0.30)	(-0.02)
+FRNNLM 4F+A	29.95	30.13	30.14
	(+0.01)	(+0.12)	(+0.13)

Table 9: *RO-ERNNLM + RO-FRNNLM: end scores*

Model	Conf1	Conf2	Conf3
FRNNLM 4F	29.99	30.19	29.99
	(+0.05)	(+0.18)	(-0.02)
FRNNLM 4F+N	29.90	30.29	30.23
	(+0.05)	(+0.28)	(+0.24)
FRNNLM 4F+A	30.00	30.20	30.21
	(+0.05)	(+0.19)	(+0.20)

In another experiment, the ERNNLM is used together with FRNNLM for rescoring. The results are illustrated in Table 9. The improvement over the sole FRNNLM 4F is indicated in parentheses. It turns out that the combined use of Wikipedia categories and Wikipedia topic information based on tf-idf performs about 0.2 BLEU better in two different configurations (Conf2 and Conf3). The FRNNLM using Wikipedia categories for all word types together with the ERNNLM achieve an improvement of 0.07 BLEU over the system without ERNNLM in these configurations. The best system exhibits a score of 30.23 BLEU, which is 0.22 BLEU better than the best system of the previous work [5] that has the same vocabulary size.

5. Conclusion

This work provides the novel idea to integrate higher-level information from encyclopedic sources, such as Wikipedia, into RNNLMs. Two approaches are proposed: the first uses a FRNNLM, the second incorporates more complex information by using a dedicated feature layer in the conventional recurrent neural model. In addition, three methods are introduced to prepare the additional data and represent it in the correct form. By using topic information from large encyclopedia, we have improved state-of-the-art translation systems on two different language pairs. This work exhibits great potential for low-resource translation tasks.

6. Acknowledgements

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement n° 645452 and the Continuous Learning in International Collaborative Studies exchange program.

7. References

[1] J. A. Bilmes and K. Kirchhoff, "Factored language models and generalized parallel backoff," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. Association for Computational Linguistics, 2003, pp. 4–6.

[2] P. Koehn and H. Hoang, "Factored translation models," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 2007, pp. 868–876.

[3] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Interspeech*, vol. 2. ISCA, 2010, p. 3.

[4] Y. Wu, X. Lu, H. Yamamoto, S. Matsuda, C. Hori, and H. Kashioka, "Factored language model based on recurrent neural network," in *Proceedings of COLING*. Association for Computational Linguistics, 2012, pp. 2835–2850.

[5] J. Niehues, T.-L. Ha, E. Cho, and A. Waibel, "Using factored word representation in neural network language models," in *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, 2016.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *SLT*. IEEE, 2012, pp. 234–239.

[8] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, p. 391, 1990.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[10] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using wikipedia-based explicit semantic analysis," in *International Joint Conferences on Artificial Intelligence Organization*, vol. 7, 2007, pp. 1606–1611.

[11] M. Strube and S. P. Ponzetto, "Wikirelate! computing semantic relatedness using wikipedia," in *American Association for Artificial Intelligence*, vol. 6, 2006, pp. 1419–1424.

[12] X. Han and L. Sun, "An entity-topic model for entity linking," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 2012, pp. 105–115.

[13] S. Cucerzan, "Large-scale named entity disambiguation based on wikipedia data," in *Proceedings of the*

2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), vol. 7. Association for Computational Linguistics, 2007, pp. 708–716.

[14] R. C. Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation," in *European Chapter of the Association for Computational Linguistics*, vol. 6, 2006, pp. 9–16.

[15] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th conference on Computational linguistics*, 1992, pp. 539–545.

[16] "汉典 zdic.net," http://www.zdic.net/, (Accessed on 09/12/2016).

[17] G. Salton and M. J. McGill, "Introduction to modern information retrieval," 1986.

[18] R. B. Bradford, "An empirical study of required dimensionality for large-scale latent semantic indexing applications," in *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 2008, pp. 153–162.

[19] "gensim: Topic modelling for humans," https://radimrehurek.com/gensim/, (Accessed on 11/22/2016).

[20] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *Advances in Neural Information Processing Systems*, 2010, pp. 856–864.

[21] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, vol. 1, 2003, pp. 160–167.