Out-of-Vocabulary Word Probability Estimation using RNN Language Model

Irina Illina, Dominique Fohr

Out-of-Vocabulary Word Probability Estimation using RNN Language Model

Irina Illina[1,2,3], **Dominique Fohr**[1,2,3]

[1] Université de Lorraine, LORIA, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
[2] Inria, Villers-les-Nancy, F-54600, France
[3] CNRS, LORIA, UMR 7503, Vandoeuvre-les-Nancy, F-54506, France
illina@loria.fr, dominique.fohr@loria.fr

Abstract

One important issue of speech recognition systems is Out-of Vocabulary words (OOV). These words, often proper nouns or new words, are essential for documents to be transcribed correctly. Thus, they must be integrated in the language model (LM) and the lexicon of the speech recognition system. This article proposes new approaches to OOV proper noun estimation using Recurrent Neural Network Language Model (RNNLM). The proposed approaches are based on the notion of closest in-vocabulary (IV) words (*list of brothers*) to a given OOV proper noun. The probabilities of these words are used to estimate the probabilities of OOV proper nouns thanks to RNNLM. Three methods for retrieving the relevant list of brothers are studied. The main advantages of the proposed approaches are that the RNNLM is not retrained and the architecture of the RNNLM is kept intact. Experiments on real text data from the website of the *Euronews* channel show perplexity reductions of about 14% relative compared to baseline RNNLM.

Keywords: Speech recognition, language modeling, OOV

1. Introduction

Voice is seen as the next big field for computer interaction. The research company Gartner reckons that by 2018, *30*% of all interactions with devices will be voice-based: people can speak up to four times faster than they can type, and the technology behind voice interaction is improving all the time. As of October 2017, *Amazon Echo* is present in about *4*% of American households. Voice assistants are proliferating in smartphones, too: Apple's Siri handles over *2* billion commands a week, and *20*% of Google searches on Android-powered handsets in America are done by voice input. Dictating e-mails and text messages works reliably enough to be useful. In this context, an automatic speech recognition system (ASR) should accommodate all voices, all topics and all lexicons.

The proper nouns (PNs) play a particular role: they are often important to understand a message and can vary enormously. For example, a voice assistant should know the names of all your friends; a search engine should know the names of all famous people and places, names of museums, etc. For the moment, it is impossible to add all existing proper nouns into a speech recognition system. A competitive approach is to dynamically add new PNs into the ASR system. It implies knowing where to look for them, and knowing how to introduce them into the lexicon and into the language model. Updating the language model of the ASR system with a list of retrieved OOV PNs is the central point of this article.

Although the LM adaptation to contextual factors (style, genre, topic) (Chen *et al.*, 2015, Wen *et al.*, 2013) has been well studied, there is little work done on integration of new words in language model. Traditionally, integration of new words is performed implicitly by using the '*unk*' word and *back-off* probability. Open vocabulary ASR represents an OOV word by a sub-lexical model (Bisani *et al*., 2005) or as sub-word units (Parada et al. 2011, Shaik *et al.* 2011). Qin (2013) proposed to estimate *n*-gram LM scores for OOV words from syntactically and semantically similar in-vocabulary (IV) words. In class-based approaches (Naptali *et al.*, 2012), an OOV is assigned to a word class and the OOV LM probability is taken from this class.

In a previous work, we proposed several approaches to estimate the bigram probability of OOV proper nouns using word similarity (Currey *et al.,* 2016). In our current work, we propose new methods for estimating OOV proper noun probability using Recurrent Neural Networks-based language model (RNNLM). The main advantage of RNNLM is a possibility of using arbitrarily long histories (Mikolov, 2013, Mikolov *et al.* 2010). Using classes at the output layer allows to speed-up the training (Mikolov, 2011). A novel aspect of the proposed methodology is the notion of *brother words*: for each OOV PN we look for a list of "similar" in-vocabulary words, called a *list of brothers*, and we use their RNNLM probabilities to estimate the OOV PN probabilities. The main advantage of our methodology is the fact that the RNNLM is not modified: no retraining of the RNNLM is needed and the RNN architecture is not modified, there are the same number of layers and the same number of nodes. The proposed method can be applied for other neural network LMs, such as Long Short-Term Memory model or Gated Recurrent Units model. Indeed, we do not modify the internal architecture of the model.

2. Proposed methodology

The naive solution for taking into account OOV PNs would consist in integrating all PNs contained in the available corpus in the lexicon and LM of the ASR. This solution is not feasible for several reasons: using corpus, like newswire or Wikipedia, will result in adding millions of OOV PNs (Qin, 2013). The ASR would become very slow. Moreover, it would increase acoustic confusability: many PNs could have pronunciations close to common names. Moreover, for instance, adding the names of all English footballers is useless to recognize a document that talks about war in Syria. In our work, we want

to add to the ASR only OOV PNs relevant to the document to be transcribed. In this article, we focus on dynamic updating of the language model.

In our methodology, we assume that we have a list of retrieved OOV proper nouns and we want to estimate their language model probability using a previously trained RNN LM. The list of OOV PNs can be retrieved according to the semantic context modeling of OOVs (Sheikh *et al.*, 2017). This list will be added to the original lexicon of ASR. In this paper, we want to integrate the list of OOVs in RNNLM using a contemporary corpus. It is important to notice that the RNNLM is ***not retrained***; it is used to estimate the probabilities of OOV words. Therefore, as inputs we have a previously trained RNNLM, the original lexicon, the list of OOV proper nouns and some text data, called *contemporary corpus*. As output, we want to estimate LM probability for OOV proper nouns using RNNLM.

We assume that the topology of RNN used for LM consists of three layers. The input layer consists of a vector $w(t)$ that represents the current word w_t encoded as *1* (size of $w(t)$ is equal to the size of the vocabulary V), and a context vector $h(t-1)$ that represents values of the hidden layer from the previous time step (see Figure 1). The output layer represents $P(w_{t+1} \mid w_t, h(t-1))$. The aim of RNNLM is to estimate the probability $P(wt+1 \mid w_t, h(t-1))$.

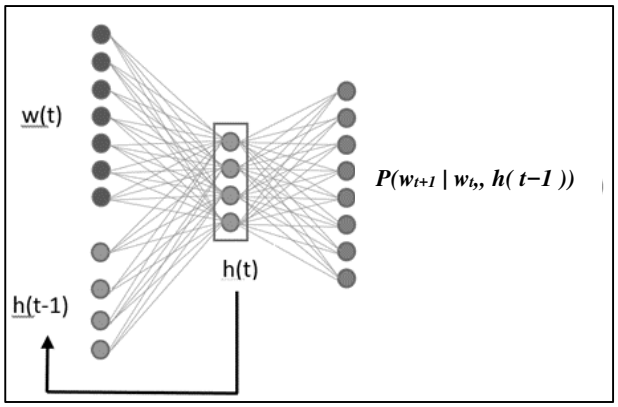

Fig. 1: Schematic representation of RNNLM.

To take into account OOV words, we have two problems:

- w_t (previous word) can be an OOV;
- or w_{t+1}(predicted word) can be OOV.

For the first case, the difficulty is how to find a relevant representation of OOV at the RNNLM input. One classical solution is to add a specific neuron for all OOVs (Sundermeyer *et al.*, 2013), but all OOVs will be treated in the same way, which is not optimal. We propose to introduce a specific representation for each OOV using the similar in-vocabulary words (*brother list*).

For the second case, we propose to estimate the probability $P(OOV \mid w_t, h(t-1))$ using the probabilities (given by the RNNLM) of the in-vocabulary words.

The main idea of our method is to build a list of similar in-vocabulary words for each OOV PN. The similarity can be at the syntactic/semantic level. It means that the in-vocabulary brother words will play the same syntactic or/and semantic role as the corresponding OOV PNs. For instance, for the OOV proper noun *Fukushima*, the brother word can be another Japan city, like *Tokyo*. The list of similar in-vocabulary words will be used to generate the input of RNNLM or to use the RNNLM output probabilities to compute probabilities for each OOV PN. The structure of the RNNLM and the weights are neither modified nor retrained.

The approaches proposed in this article include the following steps:

- Finding a list of in-vocabulary words similar to OOVs, called *list of brothers*, using a contemporary corpus (see section 2.1).
- Using the brother lists of in-vocabulary words, estimating the probabilities $P(w_{t+1}|OOV, h(t-1))$ and $P(OOV \mid w_t, h(t-1))$ for each OOV using RNNLM (see section 2.2).

In the following sections, we will present these two steps.

2.1. Brother list generation

For each OOV from the list of OOVs, we want to generate a list of size M containing ranked in-vocabulary words called *brother list*:

$$BrotherList(OOV) = \{(IV_1, v_1), (IV_2, v_2), \dots (IV_M, v_M)\} \quad (1)$$

$$g(OOV, IV_i) = v_i \quad (2)$$

where v_i corresponds to the similarity value of ith IV. Each word of this list is "similar" in some sense to the OOV PN. As similarity values, some "distance information" from in-vocabulary word to OOV can be used. All similarity values for the same OOV proper noun sum to *1* (linear combination). The brother list will be used to OOV PN probability estimation using RNNLM. We propose three approaches for the generation of the list of brothers:

- *Similarity-based approach*: to generate an IV brother list for a given OOV PN, we use a similarity measure based on *word embedding* of (Mikolov *et al.* 2013). We trained a skip-gram model with a context window size of two on a large text corpus (we assume that the OOV PN is present in this corpus). According to *word2vec*, we compute the *cos*-distance between the OOV embedding vector and the in-vocabulary embedding vectors. We choose the top M in-vocabulary words and put them in the brother list for this OOV PN. We propose to use the corresponding *cos*-distance as v_i (after normalization).
- *n-gram counting approach*: in this approach we assume that if one in-vocabulary word w occurs in the same context as that an OOV PN, then w can be used as a similar word for this OOV proper noun. To find the brother list for one OOV PN, we propose to count all k-grams $< w1, \dots w, .., wk >$ corresponding to k-grams $<w1, .., OOV, .., wk>$ where the central OOV proper noun is replaced by w, the preceding words and the following words being the same. The M central words with the highest counts will be put in the brother list for this OOV proper noun. For a small value of k (*2,3*), it is possible to find a large number of central words w. For large value of k, the number of k-grams can be very small and so, we can have few brothers.

$$class(IV) = \{IV \text{ and all OOV PNs such as this IV is in the brother list of the OOV PNs}\} \quad (5)$$

$$BrotherList(Fukushima) = \{(Tokyo, 0.6), (Nagasaki, 0.4)\} \quad (6)$$

$$BrotherList(Sendai) = \{(Tokyo, 0.5), (Nagasaki, 0.3), (Nagoya, 0.2)\} \quad (7)$$

$$class(Tokyo) = \{Tokyo, Fukushima, Sendai\} \quad (8)$$

$$class(Nagasaki) = \{Nagasaki, Fukushima, Sendai\} \quad (9)$$

$$P(Fukushima \mid w_t, h(t-1)) = P(class(Tokyo) \mid w_t, h(t-1)) * P(Fukushima \mid class(Tokyo), w_t, h(t-1)) + P(class(Nagasaki) \mid w_t, h(t-1)) * P(Fukushima \mid class(Nagasaki), w_t, h(t-1)) \quad (10)$$

$$P(Fukushima|class(Tokyo, w_t, h(t-1)) = (1-\alpha) * g(Fukushima, Tokyo)/(g(Fukushima, Tokyo) + g(Sendai, Tokyo)) \quad (11)$$

$$P(Fukushima|class(Nagasaki, w_t, h(t-1)) = (1-\alpha) * g(Fukushima, Nagasaki)/(g(Fukushima, Nagasaki) + g(Sendai, Nagasaki)) \quad (12)$$

- *Wikipedia-based approach:* we take into account only OOVs that are the last names of a person name. We assume also that the persons are famous and that a Wikipedia page exists for them. In this aim, we have collected all Wikipedia webpage titles. For an OOV word, we search for all titles of Wikipedia containing this OOV. From these titles, we choose all fist names of this OOV word. After this, we search all last names of these first names from Wikipedia titles and put them in the brother list for this OOV. For instance, for OOV word *Kaymer* we find the title webpage *Martin Kaymer* (professional golfer). Then we search for webpage titles with *Martin* as first name and we find *Martin Scorsese*, *Martin Luther*, *Martin Malvy*, etc. Therefore, the brother list of the OOV word "*Kaymer*" will contain *Scorsese*, *Luther*, *Malvy*, etc.

2.2. OOV PN probability estimation using RNNLM

For computing the probability of a sentence containing OOV PNs, we propose to use the brother list of each OOV PN.

2.2.1. Computing P (w_{t+1} | OOV, h (t−1))

As OOV proper noun is not in the lexicon, RNNLM has no corresponding input neuron for it. We propose to represent each OOV PN by a linear combination of in-vocabulary words from the brother list of this OOV. For instance, if the brother list of an OOV proper noun contains *2* IVs:

$$BrotherList(OOV) = \{(IV_1, 0.6), (IV_2, 0.4)\} \quad (3)$$

The RNN input vector for this OOV proper noun will be:

$$w(t) = (0 \ldots 0 \ \ 0.6 \ \ 0 \ \ldots \ 0 \ \ 0.4 \ \ 0 \ldots 0) \quad (4)$$

where *0.4* and *0.6* correspond to the similarity values of two IV words and their positions (instead of a single *1* in a classical RNN). In this case, the OOV can be seen as a linear combination of IV words of the brother list. If brother list contains *M* words, all *M* in-vocabulary words can be used. After this, the input is propagated through the RNNLM. At the output, we will obtain probabilities $P(w_{t+1} \mid OOV, h(t-1))$.

2.2.2. Computing P(OOV | w_t,, h(t−1))

As OOV PN is not in the lexicon, RNNLM has no corresponding output neuron for it. The probability of OOV will be estimated using the probabilities of in-vocabulary proper noun from the brother list. For each IV, we define a *class* containing the in-vocabulary word itself and all OOV proper nouns for which this IV is a brother (cf. Eq. (5)).

As an example, let us consider that we have two OOVs: *Fukushima* and *Sendai.* The obtained brothers for *Fukushima* are the IVs *Tokyo* and *Nagasaki* (cf. Eq. (6)). The obtained brothers of *Sendai* are the IVs *Tokyo, Nagasaki* and *Nagoya* (cf. Eq. (7)). We can define the classes of *Tokyo* and *Nagasaki* according to Eq. (8) and (9). We compute the probability of OOV PN *Fukushima P(Fukushima| w_t, h(t−1))* as defined by Eq. (10). *P(class(Tokyo)|w_t, h(t−1))* and *P(class(Nagasaki)|w_t, h(t−1))* are computed by the RNNLM.

We can compute $P(Fukushima|class(Tokyo, w_t, h(t-1)))$ and $P(Fukushima|class(Nagasaki, w_t, h(t-1)))$ according to Eq. (11) and (12). α represents the proportion of probability mass that we put on the IV of *class(IV). (1- α)* represents the proportion of probability mass that we put on the OOV of *class(IV).* This weight is adjusted experimentally. It should be possible to have one α per *class(IV)*, but it would be difficult to accurately estimate these parameters. We chose to estimate only one α for all words.

$$P(Tokyo|w_t, h(t-1)) = P(class(Tokyo)) * \alpha \quad (13)$$

This ensures that the sum of probability of all words is one:

$$\sum_{m \in IV} P(m|w_t, h(t-1)) + \sum_{m \in OOV} P(m|w_t, h(t-1)) = 1$$

3. Experimental setup

3.1. Data description

3.1.1. Training textual corpora

We used the following corpora for training our language model, OOV PN retrieval system and brother list's generation:

- *Le Monde:* textual data from the French newspaper *Le Monde* (*200*M words; corresponding to 1988-2006, only even years)
- *Le Figaro*: textual data from the French newspaper *Le Figaro* (*8*M words, 2014)
- *L'Express*: textual data from the French newspaper *L'Express* (*51*M words, 2014).

The original LM was trained using the *Le Monde* corpus. The lists of OOV PNs to add were created using the *l'Express* corpus. The *Le Figaro+l'Express* corpus was used as the *contemporary corpus* for estimating word embedding and for generating brother lists. These corpora correspond to the same time period as the development and test data. Such corpus could be retrieved from the Internet.

3.1.2. Development and test textual corpus

The development and test corpus come from the website of the Euronews channel: textual news articles from January 2014 to

June 2014 (Sheikh et al. 2016). We selected only the sentences containing at least one OOV word. For the development and test, we used the same number of sentences *1148* sentences (about *29*K words per corpus, different sets of sentences for development and test corpus). The development corpus is used to evaluate the methodology proposed in this paper and to adjust the involved parameters. The test corpus is used to evaluate the results using adjusted parameters. The results will be presented in term of *word perplexity*.

3.1.3. Test audio corpus

The test audio corpus consists of video files reports from the *Euronews* website and their accompanying transcripts (2014). It could be noted, that the reference transcriptions for the recognition experiments are the transcripts provided with the news videos, which may not always be an exact match to the audio. The test audio corpus consists of 300 articles (60K words) and the OOV rate is about 2%. The number of retrieved OOV PNs is 9300 OOVs. Confidence interval is ± 0.3%.

3.2. RNNLM

The lexicon contains about *87*K words. For RNNLM we used the toolkit developed by Mikolov (Mikolov *et al.,* 2011a) with *310* classes and *500* hidden nodes. The standard backpropagation algorithm with stochastic gradient descent is used to train the network.

3.3. OOV proper noun list

The ***original lexicon*** of *87*k words is augmented by adding the retrieved OOV proper noun word list as follows:

- For each development/test file, we create a ranked list of OOV proper nouns according to the methodology presented in (Sheikh *et al.,* 2017);
- From each list we keep only top *128* words;
- All lists from the development set are merged into one list; all lists from the test set are merged into one list.

Finally, we obtain the ***extended lexicon*** of 95K words.

3.4. Language model

In our experiments, different language models are used. It is important to notice that ***all language models contain the same vocabulary***: the extended lexicon (95K words).

- **The baseline** RNNLM language model is built as follows: it is trained using the original lexicon (87K words) on the train corpus (*Le Monde* corpus). The probability of an OOV from the retrieved OOV proper noun list is computed using the probability of *unk* estimated by the RNNLM. We consider *unk* as a class corresponding to all OOV proper noun words:

$$P(OOV \mid w_t, h(t-1)) = P(class(unk)) * P(OOV|class(unk)) \quad (14)$$

where *P(class(unk))* is computed by the RNN (output neuron corresponding to *unk*). To estimate *P(OOV | class(unk)),* we assume that all OOVs are equiprobable:

$$P(OOV \mid class(unk)) = 1/NbrOOV_{train} \quad (15)$$

where $NbrOOV_{train}$ is the number of OOV PNs in the training corpus. A similar approach was used in (Sundermeyer *et al.*, 2013).

- **The modified** RNNLM is the same as the baseline LM and corresponds to the extended lexicon, but the probabilities are estimated according to the proposed methodology.

Note that these LMs *have* ***the same number of words***, corresponding to the extended lexicon, and so the computed perplexities will be comparable.

During brother generation, we removed stop words (articles, adverbs, adjectives) from the brother list, because it is unlikely that these words appear in the same context as the proper nouns. So they cannot be used as brother words.

4. Results

As usual, the development corpus is used to tune the parameters and to find the best configuration for each method. After this, the best configuration is evaluated on the test corpus.

4.1. Results on the development corpus

Table 1 gives examples of brother list generation for some OOVs using similarity-based and Wikipedia-based approaches.

OOVs	Brother words
Similarity-based approach	
CEZ	*Microsoft, KPN, Vivendi*
Bouar	*Donetsk, Kidal, Kharkiv, Kayes, Tripoli, Lucerne, Brno, Paris, Bamako, Gaza*
Randstad	*Areva, CNPC, Dassault, Boursorama, MSF, Dongfeng, Ikea, Airbus, Bercy*
Kaymer	*Andre, Martin, Citroen, Nestle*
Heslov	*Bollore, Nestle, Lagardere, Kevin*
Wikipedia-based approach	
Kaymer	*Scorsese, Luther, Malvy, Bouygues, Bangemann, Marietta, Walser, Heidegger*
Heslov	*Dalton, Fox, Hackett, Hill, Wood*

Table 1: Examples of brother lists for some OOV words using similarity-based and Wikipedia-based approaches.

We can observe that the brother choice seems to be reasonable. We would like to note, that the brother lists generated by these methods are different because the brother choice criterions are different. For example, for OOV *Kaymer,* similarity-based method proposes four words (*Andre, Martin, Citroen, Nestle)* chosen according to Mikolov similarity. While Wikipedia-based method proposes (*Scorsese, Luther, Malvy, Bouygues, Bangemann, Marietta, Walser, Heidegger, Hirsch, Winckler)* because these family names have the same first name *Martin*, as OOV name *Kaymer.*

4.1.1. Parameter choice

Figure 2 shows the evolution of the word perplexity in function of the brother number for the similarity-based approach. This number represents the maximal size of every brother list and corresponds to *M* (it is possible to have less brothers that this number). This number of brothers is used to compute $P(w_{t+1} \mid OOV, h(t-1))$ (see section 2.2.1) and $P(OOV \mid w_t, h(t-1))$ (see section 2.2.2). We can observe that using only one or two brothers gives a high word perplexity.

Using more brothers is better. The best value of the brother number is around 26 brothers for similarity-based approach. In the following experiments, we will use 26 brothers for this approach. For *n*-gram counting approach, the best value is 28 and for Wikipedia-based approach 5 is optimal.

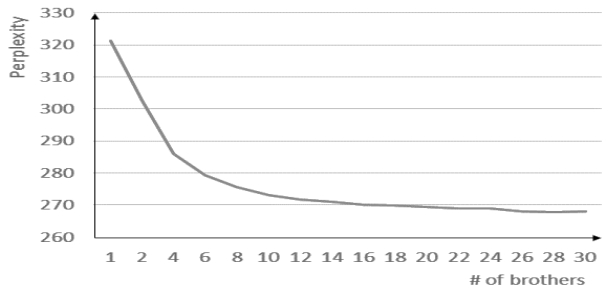

Fig. 2: Perplexity versus maximal size of every brother list (*M*) for similarity-based approach. Development text corpus, *α=0.6.*

Figure 3 presents the word perplexity evolution in function of the coefficient *α* (cf. Eq. (11) - (13)) for similarity-based approach. *(1 - α)* can be seen as the probability mass that is removed from the IV words to be given to the OOV words. The perplexity decreases when coefficient *α* increases until 0.6. After this value, the perplexity begins to increase. We decided to use this value of 0.6 for this method in the following experiments This means that for this method the probability mass that we put on the IV of *class(IV)* is 0.6. For other brother generation methods this coefficient is adjusted experimentally, method per method.

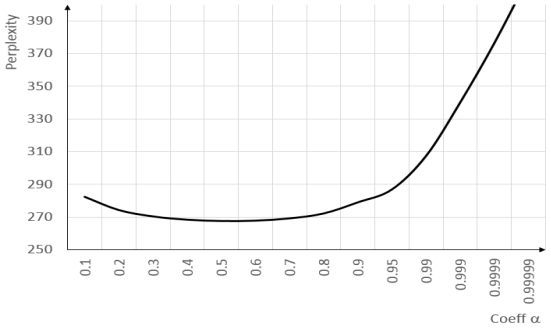

Fig. 3: Perplexity versus coefficient *α* for similarity-based approach. Development text corpus, brother number (*M*) is 26.

4.1.2. Word perplexity results

Table 2 presents the perplexity results of experiments on the development data. In this table, as previously, *#brothers* represents the maximal size of every brother list and corresponds to *M*. It is important to note that in these experiments extended lexicon is used. For *n*-gram brother generation method, larger context (*k = 5*) gives a better result than the smaller context (*k = 3*): larger context contains more information about the similarity between IV and OOV words.

The best result is obtained by similarity-based method: we obtained the perplexity of 267.9 compared to the perplexity of 311.4 for baseline method. We note an important difference between the results of two brother generation methods: PPL of 267.9 for similarity-based and 299.0 for *n*-gram-based methods. This can be explained by the fact that Mikolov's word embedding allows to better model the word contexts. We tried to mix the two best approaches, but no word perplexity improvement was observed.

In conclusion, from this table we observe that the proposed method for OOV integration in the RNNLM using similarity-based brother generation gives a good perplexity reduction over the baseline: the reduction is *14*% for the best configuration, compared to the baseline RNNLM (267.9 versus 311.4).

Language models	**# Brothers *(M)***	***α***	**PPL**
Baseline RNNLM			311.4
Modified RNNLM, similarity-based	26	0.6	**267.9**
Modified RNNLM, *n*-gram counting, *k=5*	28	0.9	299.0
Modified RNNLM, Wikipedia-based	5	0.9	295.5

Table 2: Word perplexity results for OOV proper noun's probability estimation in the RNNLM on the development text corpus.

4.2. Results on the test text corpus

The best-performing configuration of brother selection methods from the experiments on the development data is applied to the test data. For similarity-based brother selection method, we use the list of 26 brothers and *α=0.6.* For *n*-gram based brother selection method, we use a list of 28 brothers with *α = 0.9, k = 5.*

Table 3 displays the word perplexity results on the test data. The results are consistent with the results obtained on the development data. The proposed methods improve the perplexity compared to the baseline system. As previously, *n*-gram count and Wikipedia-based methods perform worse than the similarity-based method. The best perplexity reduction is *14* % relative compared to the baseline RNNLM (258.6 versus 299.5). This improvement is consistent to the one obtained on the development set.

Language models	**PPL**
Baseline RNNLM	299.5
Modified RNNLM, similarity-based, *26* brothers, *α=0.6*	**258.6**
Modified RNNLM, *n*-gram count, *k=5*, *28* brothers, *α= 0.9*	291.4
Modified RNNLM , Wikipedia-based, *k=5*, *28* brothers, *α=0.9*	283.2

Table 3: Word perplexity results for OOV proper noun's probability estimation in the RNNLM on the test text corpus.

4.3. Recognition results on the test audio corpus

After finding the best parameters and algorithms on the text corpus, we use the test audio corpus to further examine speech recognition system performance.

The Kaldi-based Automatic Transcription System (KATS) uses context dependent DNN-HMM phone models. These models are trained on 250-hour broadcast news audio files. Using the SRILM toolkit (Stolcke *et al.* 2011), a pruned bigram and trigram language models are estimated on the *Le*

Monde + Gigaword corpus and used to produce the word lattice. From lattice, we extracted 200-best hypotheses and we rescored them with the RNNLMs (baseline RNNLM and modified RNNLM using similarity-based approach).

We computed the Word Error Rate (WER) for three language models: RNNLM with original lexicon (87K words); baseline RNN language model with extended lexicon (95K words); modified RNNLM using similarity-based approach with the best parameter set and using extended lexicon (95K words). Last two RNNLM correspond to the models used in the previous sections. All these models are used to rescore 200-best hypotheses.

Lexicons and language models	WER(%)
Original lexicon and rescoring with baseline RNNLM	20.2
Extended lexicon and rescoring with baseline RNNLM	**18.7**
Extended lexicon and rescoring with modified RNNLM, similarity-based, 26 brothers, $\alpha=0.6$	**18.7**

Table 4: WER results using different lexicons and RNN language models on the audio corpus.

The results for the recognition experiments on the audio corpus are shown in Table 4. Baseline RNNLM with original lexicon gives 20.2% WER. Using extended lexicon with baseline RNNLM or with modified RNNLM gives similar results: 18.7%. Thus, extended lexicon yielded a statistically significant improvement over the original lexicon. In contrast, no improvement is observed for the proposed method (18.7% WER) compared to baseline RNNLM with extended lexicon. However, proposed similarity-based method obtained a good perplexity improvement compared to the baseline RNNLM on the development and test corpus (cf. section 4.1 - 4.2). This can be due to the fact that reducing perplexity does not always imply a reduction of WER.

5. Conclusion

In this paper, we explore different ways of adding OOVs to the language model of ASR. We propose new approaches to OOV proper noun probability estimation using RNN language model. The key ideas are to use similar in-vocabulary words, word-similarity measures, *n*-gram counting and Wikipedia. The main advantage of our methodology is that the RNNLM is not modified and no retraining or adaptation of the RNNLM is needed. The proposed methods can be applied for other NN LMs (more hidden layers or LSTM/GRU layers), because we do not modify the internal architecture of the model.

Experimental results show that the proposed approaches achieve a good improvement in word perplexity over the baseline RNNLM system, and that the similarity-based approach gives the lowest perplexity among all.

In state-of-the-art ASR systems, RNNLMs are often linearly interpolated with *n*-gram LMs to obtain both a good context coverage and effective generalization. The investigation to integrate the *n*-gram model in our Kaldi-based speech recognition system (as proposed in (Currey *et al.,* 2016) and the methods proposed in this article is a future work.

Acknowledgements

This work is funded by the *ContNomina* project supported by the French National Research Agency (ANR) under contract ANR-12-BS02-0009. Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria, CNRS, RENATER and other Universities and organizations (https://www.grid5000.fr).

References

Bisani, M., Ney, H. (2005). *Open Vocabulary Speech Recognition with Flat Hybrid Models*. In Proc. of Interspeech.

Chen, X., Tan, T., Liu, X., Lanchantin, P., Wan, M., Gales, M.J.F., Woodland, P.C. (2015). *Recurrent Neural Network Language Model Adaptation for Multi-Genre Broadcast Speech Recognition.* In Proc. of Interspeech.

Currey, A., Illina, I., Fohr, D. (2016). *Dynamic Adjustment of Language Models for Automatic Speech Recognition using Word Similarity*. In Proc. of IEEE Workshop on Spoken Language Technology (SLT).

Mikolov, T. (2013). *Statistical Language Models Based on Neural Networks*. Ph. D. thesis, Brno University of Technology.

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., Khudanpur, S. (2010). *Recurrent Neural Network based Language Lodel.* In Proc. of Interspeech.

Mikolov, T., Kombrink, S., Burget, L., Cernocky, J., Khudanpur, S. (2011). *Extensions of Recurrent Neural Network Language Model.* In Proc.of IEEE ICASSP, pp. 5528–5531.

Mikolov, T., Kombrink, S., Deoras, A., Burget, L., Cernocky, J. (2011a). *RNNLM - Recurrent Neural Network Language Modeling Toolkit*. In Proc. of IEEE ASRU.

Mikolov, T., Sutskever, I., Chen, K., Corrado, S., Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. In Advances in Neural Information Processing Systems, pp. 3111-3119.

Naptali, W., Tsuchiya, M., Nakagawa, S. (2012). *Class-Based N-Gram Language Model for New Words Using Out-of-Vocabulary to In-Vocabulary Similarity*. In IEICE Transactions on Information and Systems, Vol. E95-D, No. 9, pp. 2308-2317.

Parada, C., Dredze, M., Sethy, A., Rastrow, A. (2011). *Learning Sub-Word Units for Open Vocabulary Speech Recognition.* In Proc. of ACL.

Qin, L. (2013). *Learning Out-of-Vocabulary Words in Automatic Speech Recognition.* Ph. D. thesis, CMU University.

Shaik, A., El-Desoky Mousa, Schlüter, R; Ney, H. (2011). *Hybrid Language Models Using Mixed Types of Sub-Lexical Units for Open Vocabulary German LVCSR*. In Proc.of Interspeech, pp. 1441-1444.

Sheikh, I. Fohr, D., Illina, I., Linares, G. (2017). *Modelling Semantic Context of OOV Words in Large Vocabulary Continuous Speech Recognition.* In IEEE/ACM Transactions on Audio, Speech and Language Processing, N. 25 (3), pp. 598 – 610.

Sheikh, I., Illina, I., Fohr, D. (2016). *How Diachronic Text Corpora Affect Context based Retrieval of OOV Proper Names for Audio News.* In Proceedings of LREC, Portoroz, Slovenia

Stolcke, A., Zheng, J., Wang, W., Abrash V. (2011). *SRILM at sixteen: Update and outlook.* In Proc. of IEEE Automatic Speech Recognition and Understanding Workshop.

Sundermeyer, M., Oparin, I., Gauvain, J.-L., Freiberg, B., Schlüter, R., Ney, H. (2013). *Comparison of Feedforward and Recurrent Neural Network Language Models.* In Proc. of IEEE ICASSP.

Wen, T.-H., Heidel, A., Lee, H.-Y., Tsao, Y., Lee, L.-S. (2013). Recurrent Neural Network based Personalized Language Modeling by Social Network Crowdsourcing. In Proc. of Interspeech.