RNN Language Model Estimation for Out-of-Vocabulary Words

Irina Illina(✉) and Dominique Fohr

MultiSpeech Team, Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France
illina@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 probability 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 relative perplexity reductions of about 14% compared to baseline RNNLM.

Keywords: Speech recognition · Neural networks · Vocabulary extension · Out-of-vocabulary words · Proper names

1 Introduction

Voice is seen as the next big field for computer interaction. From *Statista Research Department*, as of 2019, there are an estimated 3.25 billion digital voice assistants being used in devices around the world. Global smart speaker sales hit a record high in 2019 with shipments of 146.9 million units, up 70% over 2018, according to a recent report on the state of the smart speaker market from *Strategy Analytics*. Google reports that 27% of the online global population is using voice search on mobile.

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) [2, 17] 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 [1] or as sub-word units [10, 12]. [11] proposed to estimate n-gram LM scores for OOV words from syntactically and semantically similar in-vocabulary (IV) words. In class-based approaches [9], an OOV is assigned to a word class and the OOV LM probability is taken from this class.

In our previous works, we proposed several approaches to estimate the bigram probability of OOV proper nouns using word similarity [3]. 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 [5, 8]. Using classes at the output layer allows to speed-up the training [6]. ***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 [11]. The ASR would become very slow. Moreover, it would increase acoustic confusability: many PNs could have pronunciations close to common names. 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 [13]. 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 Fig. 1). The output layer represents $P(w_{t+1}|w_t, h(t-1))$. The aim of RNNLM is to estimate the probability $P(w_{t+1}|w_t, h(t-1))$.

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 [16], 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 $P(OOV|w_t, h(t-1))$ using the probabilities (given by the RNNLM) of the in-vocabulary words of the brother list.

The main idea of our method is to build a list of similar in-vocabulary words for each OOV PN. The similarity can be modeled 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 Sect. 2.1).
- Using the brother lists of in-vocabulary words, estimating the probabilities $P(w_{t+1}|OOV, h(t-1))$ and $P(OOV|w_t, h(t-1))$ for each OOV using RNNLM (see Sect. 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 a ranked in-vocabulary words called *brother list:*

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

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

where v_i corresponds to the similarity value of *i*th 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 a given OOV proper noun sum to *1* (linear combination). The brother list will be used to estimate the OOV PN probability thanks to the 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 *word2vec* [8]. 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).
- *k-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 $<w_1, \ldots w, \ldots, w_k>$ corresponding to k-grams $<w_1, \ldots, OOV, \ldots, w_k>$ where the central OOV proper noun is replaced by w. The preceding words and the following words being the same. The N 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.
- *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.

Computing $P(w_{t+1} \mid OOV, h(t-1))$

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

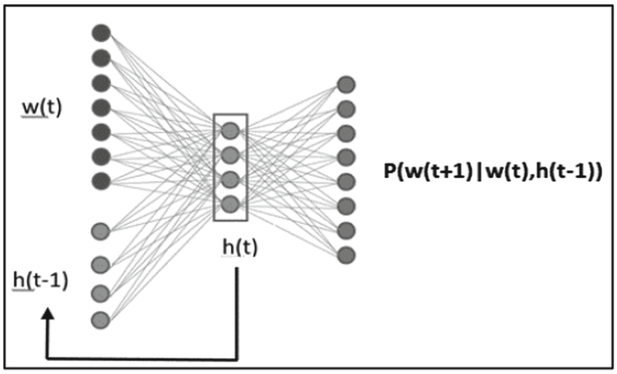

Fig. 1. Schematic representation of RNNLM.

words from the brother list of this OOV. For instance, if the brother list of an OOV proper noun contains *2* IVs:

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

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

$$w(t) = (0 \ldots 0\ \ 0.6\ \ 0 \ldots 0\ \ 0.4\ \ 0 \ldots 0) \tag{4}$$

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 one-hot representation). 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 $P(w_{t+1} | OOV, h(t-1))$.

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

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

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

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

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

$$\begin{aligned} P(Fukushima \mid w_t, h(t-1)) = {} & P(class(Tokyo) \mid w_t, h(t-1)) * P(Fukushima \mid class(Tokyo), w_t, h(t-1)) \\ & + P\left(class(Nagasaki) \mid w_t, h(t-1)\right) * P\left(Fukushima \mid class(Nagasaki), w_t, h(t-1)\right) \end{aligned} \tag{10}$$

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

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

Computing $\boldsymbol{P(OOV \mid w_t, h(t-1))}$

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 nouns 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.

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

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

$$\sum_{m \in IV} P(m|w_t, h(t-1)) + \sum_{m \in OOV} P(m|w_t, h(t-1)) = 1 \tag{14}$$

3 Experimental Setup

3.1 Data Description

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 eleven 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 embeddings and for generating brother lists. These corpora correspond to the same time period as the development and test data.

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 [14]. 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 evaluation is performed on the test corpus using the adjusted parameters. The results will be presented in term of *word perplexity.*

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. The RNNLM is trained with the toolkit developed by Mikolov [7] 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 [13];
- 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 *95*K words.

3.4 Language Model

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

- The baseline RNNLM language model is built as follows: it is trained using the original lexicon (*87*K 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* (unknown word) estimated by the RNNLM. We consider *unk* as a class corresponding to all OOV proper noun words.

$$P(OOV|w_t, h(t-1)) = P(class(unk)) * P(OOV|class(unk)) \quad (15)$$

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:

$$P(OOV|class(unk)) = 1/NbrOOV_{train} \tag{16}$$

where $NbrOOV_{train}$ is the number of OOV PNs in the training corpus. A similar approach was used in [16].

- **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. We can observe that the brother choice seems to be reasonable. We would like to note, that the brother lists generated by these methods

Table 1. Examples of brother list generation for some OOV words 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*
Randstad	*Areva, CNPC, Dassault, Boursorama, MSF, Dongfeng, Ikea*
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*

are different because the brother choice criterions are different. For example, for OOV *Kaymer,* similarity-based method proposes 4 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.*

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 $P\ (w_{t+1} \mid OOV,\ h(t-1))$ and $P(OOV \mid w_{t},,\ h(t-1))$. We can observe that using only 1 or 2 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.

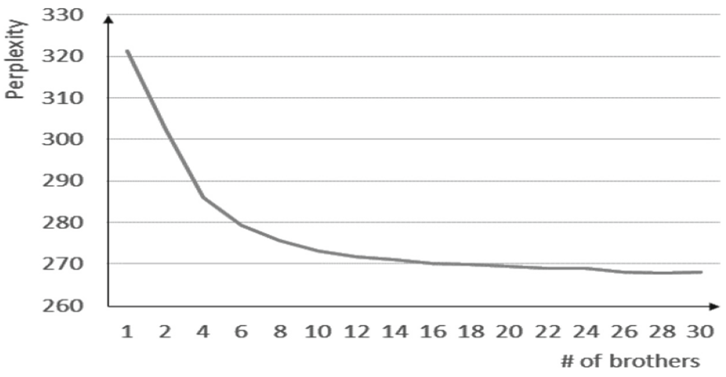

Fig. 2. Perplexity versus maximal size of every brother list (M) for similarity-based approach. Development text corpus, $\alpha = 0.6$.

Figure 3 presents the word perplexity evolution in function of the coefficient α (cf. Eq. (11)–(13)) for similarity-based approach. $(1 - \alpha)$ 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.

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 the extended lexicon

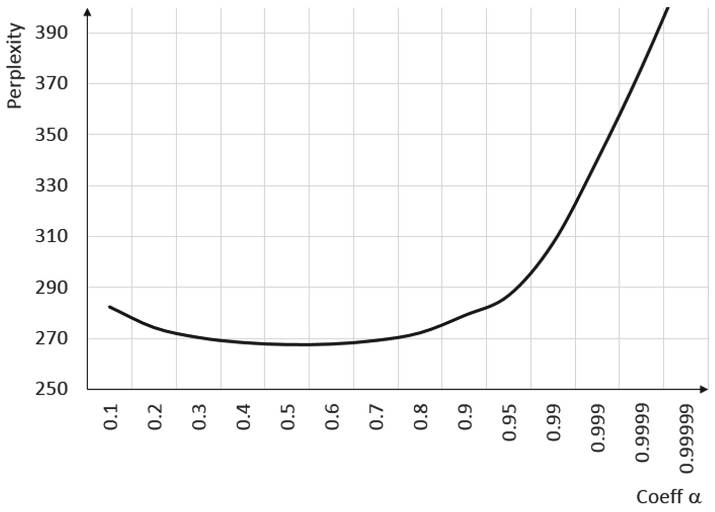

Fig. 3. Perplexity versus coefficient α for similarity-based approach. Development text corpus, brother number (M) is 26.

is used. For the k-gram brother generation method, a larger context ($k = 5$) gives a better result than a smaller context ($k = 3$): a larger context contains more information about the similarity between IV and OOV words.

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

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

The best result is obtained by the similarity-based method: we obtained the perplexity of *267.9* compared to the perplexity of *311.4* for the baseline method. We note an important difference between two brother generation method results: 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*).

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, $\alpha = 0.9$, $k = 5$.

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.

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

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

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-h broadcast news audio files. Using the SRILM toolkit [15], a pruned trigram language model is 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 (*87*K words); baseline RNN language model with extended lexicon (*95*K words); modified RNNLM using similarity-based approach with the best parameter set and using extended lexicon (*95*K words). The last two RNNLM correspond to the models used in the previous sections. All these models are used to rescore *200*-best hypotheses.

The results for the recognition experiments on the audio corpus are shown in Table 4. The baseline RNNLM with original lexicon gives *20.2%* WER. Using the extended lexicon with the baseline RNNLM or with the modified RNNLM gives similar results: *18.7%*. Thus, extended lexicon yielded a statistically significant improvement over the original

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

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**

lexicon. In contrast, no improvement is observed for the proposed method (*18.7%* WER) compared to the baseline RNNLM with the extended lexicon. However, the proposed similarity-based method obtained a good perplexity improvement compared to the baseline RNNLM on the development and test corpus (cf. Sect. 4.1 and 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.

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

1. Bisani, M., Ney, H.: Open vocabulary speech recognition with flat hybrid models. In: Proceedings of Interspeech (2005)
2. Chen, X., et al.: Recurrent neural network language model adaptation for multi-genre broadcast speech recognition. In: Proceedings of Interspeech (2015)
3. Currey, A., Illina, I., Fohr, D.: Dynamic adjustment of language models for automatic speech recognition using word similarity. In: Proceedings of IEEE Workshop on Spoken Language Technology (SLT) (2016)
4. Mikolov, T.: Statistical language models based on neural networks. Ph. D. thesis, Brno University of Technology (2013)

5. Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., Khudanpur, S.: Recurrent neural network based language lodel. In: Proceedings of Interspeech (2010)
6. Mikolov, T., Kombrink, S., Burget, L., Cernocky, J., Khudanpur, S.: Extensions of recurrent neural network language model. In: Proceedings of IEEE ICASSP, pp. 5528–5531 (2011)
7. Mikolov, T., Kombrink, S., Deoras, A., Burget, L., Cernocky, J.: RNNLM - recurrent neural network language modeling toolkit. In: Proceedings of IEEE ASRU (2011)
8. Mikolov, T., Sutskever, I., Chen, K., Corrado, S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
9. Naptali, W., Tsuchiya, M., Nakagawa, S.: Class-based N-gram language model for new words using out-of-vocabulary to in-vocabulary similarity. IEICE Trans. Inf. Syst. **E95-D**(9), 2308–2317 (2012)
10. Parada, C., Dredze, M., Sethy, A., Rastrow, A.: Learning sub-word units for open vocabulary speech recognition. In: Proceedings of ACL (2011)
11. Qin, L.: Learning Out-of-Vocabulary Words in Automatic Speech Recognition. Ph. D. thesis, CMU University (2013)
12. Shaik, A., Mousa, E.-D., Schlüter, R, Ney, H.: Hybrid language models using mixed types of sub-lexical units for open vocabulary German LVCSR. In: Proceedings of Interspeech, pp. 1441–1444 (2011)
13. Sheikh, I., Fohr, D., Illina, I., Linares, G.: Modelling semantic context of OOV words in large vocabulary continuous speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. Inst. Electr. Electron. Eng. N **25**(3), 598–610 (2017)
14. Sheikh, I., Illina, I., Fohr, D.: How diachronic text corpora affect context based retrieval of OOV proper names for audio news. In: Proceedings of LREC (2016)
15. Stolcke, A., Zheng, J., Wang, W., Abrash V.: SRILM at sixteen: update and outlook. In: Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (2011)
16. Sundermeyer, M., Oparin, I., Gauvain, J.-L., Freiberg, B., Schlüter, R., Ney, H. Comparison of Feedforward and recurrent neural network language models. In Proceedings of IEEE ICASSP (2013)
17. Wen, T.-H., Heidel, A., Lee, H.-Y., Tsao, Y., Lee, L.-S.: Recurrent neural network based personalized language modeling by social network crowdsourcing. In: Proceedings of Interspeech (2013)