Recent improvements on error detection for automatic speech recognition

Yannick Estève and **Sahar Ghannay** and **Nathalie Camelin**[1]

Abstract.
Automatic speech recognition(ASR) offers the ability to access the semantic content present in spoken language within audio and video documents. While acoustic models based on deep neural networks have recently significantly improved the performances of ASR systems, automatic transcriptions still contain errors. Errors perturb the exploitation of these ASR outputs by introducing noise to the text. To reduce this noise, it is possible to apply an ASR error detection in order to remove recognized words labelled as errors.

This paper presents an approach that reaches very good results, better than previous state-of-the-art approaches. This work is based on a neural approach, and more especially on a study targeted to acoustic and linguistic word embeddings, that are representations of words in a continuous space.

In comparison to the previous state-of-the-art approach which were based on Conditional Random Fields, our approach reduces the classification error rate by 7.2%.

1 Introduction

The advancement in the speech processing field and the availability of powerful computing devices have led to better performance in the speech recognition domain. However, recognition errors are still unavoidable, whatever the quality of the ASR systems. This reflects their sensitivity to the variability: the acoustic environment, speaker, language styles and the theme of the speech. These errors can have a considerable impact on the application of certain automatic processes such as information retrieval, speech to speech translation, etc.

The encountered errors can be due to a misinterpretation of the signal. For example, the noise associated with the sound of the environment or a problem with the quality of recording channel is interpreted as speech by the system. One of the source of errors may also come from a mispronunciation of a word, a non respect speech turn when two speakers are involved at the same time also creates a disturbance of the sound signal.

The efficient generation of speech transcriptions in any condition (*e.g.* noise free environment, etc.) remains the ultimate goal, which is not already solved. Error detection can help to improve the exploitation of ASR outputs by downstream applications, but is a difficult task given the fact that there are several types of errors, which can range from the simple substitution of a word with a homophone to the insertion of an irrelevant word for the overall understanding of the sequence of words. They can also affect neighboring words and create a whole area of erroneous words.

[1] LIUM - University of Le Mans, France, email: name.surname@univ-lemans.fr

Error detection can be performed in three steps: first, generating a set of features that are based on ASR system or gathered from other source of knowledge. Then, based on these features, estimating correctness probabilities (confidence measures). Finally, a decision is made by applying a threshold on these probabilities.

Many studies focus on the ASR error detection. In [14], authors have applied the detection capability for filtering data for unsupervised learning of an acoustic model. Their approach was based on applying two thresholds on the linear combination of two confidence measures. The first one, was derived from language model and takes into account backoff behavior during the ASR decoding. This measure is different from the language model score, because it provides information about the word context. The second is the posterior probability extracted from the confusion network. In [5], authors addressed the issue of error region detection and characterization in Large Vocabulary Continuous Speech Recognition (LVCSR) transcriptions. They proposed to classify error regions in four classes, in particular, they are interested in a person noun error which is a critical information in many information retrieval applications. They proposed several sequential detection, classification approaches and an integrated sequence labeling approach. The ASR error detection problem is related to the Out Of Vocabulary (OOV) detection task, considering that OOV errors behavior and impact differ from other errors, assuming that OOV words contribute to recognition errors on surrounding words. Many studies focused on detecting OOV errors. More recently, in [17], authors have also focused on detecting error regions generated by OOV words. They proposed an approach based on CRF tagger, which takes into account contextual information from neighboring regions instead of considering only the local region of OOV words. This approach leads to significant improvement compared to state of the art. The generalization of this approach for other ASR errors was presented in [1], which proposes an error detection system based on CRF tagger using various ASR, lexical and syntactic features. Their experiments that are performed on two corpora in English for the DARPA BOLT project showed the validity of this approach for the detection of important errors. In [18], new features gathered from other knowledge sources than the decoder itself were explored for ASR error detection, which are a binary feature that compares the outputs from two different ASR systems (word by word), a feature based on the number of hits of the hypothesized bigrams, obtained by queries entered into a very popular Web search engine, and finally a feature related to automatically infered topics at sentence and word levels. Two out of three new features, a binary word match feature and a bigram hit feature, led to significant improvements, with a maximum entropy model and CRF with linear-chain conditional random fields, comparing to a baseline using only decoder-based features. A neural network classifier trained to locate errors in an utterance using a variety of features is presented in [20]. Two approaches are proposed to extract confidence measures : the first one, is based on Recurrent Neural network Language Model (RNNLM) features to capture long-distance context within and across previous utterances. The second one, consist of combining complementary state-of-the-art DNN and GMM ASR for effective error detection, by leveraging DNN and GMM confusion networks that store word confusion information from multiple systems for feature extraction.

The ASR error detection method presented in this paper is based on incorporating a set of features in the confidence classifier built on neural network architectures, including MLP and DNN, which is in charge to attribute a label (error or correct) for each word of an ASR hypothesis.

A combination approach based on the use of an auto encoder is applied to combine well-known word embeddings: this combination helps to take benefit from the complementarities of these different word embeddings, as recently shown in one of our previous studies [10].

2 ASR error detection based on word embeddings

The error detection system has to attribute the label *correct* or *error* to each word in the ASR transcript. Each decision is based on a set of heterogeneous features. In our approach, this classification is performed by analyzing each recognized word within its context.

The proposed ASR error detection system is based on a feed forward neural network and is designed to be fed by different kinds of features, including word embeddings.

2.1 Architecture

This ASR error detection system is based on a multi-stream strategy to train the network, named multilayer perceptron multi stream (MLP-MS). The MLP-MS architecture is used in order to better integrate the contextual information from neighboring words. This architecture is inspired by [7] where word and semantic features are integrated for topic identification in telephone conversations. The training of the MLP-MS is based on pre-training the hidden layers separately and then fine tuning the whole network. The proposed architecture, depicted in Figure 1, is detailed as follows: three feature vectors are used as input to the network – feature vectors are described in the next section. These vectors are respectively the feature vector representing the two left words (L), the feature vector representing the current word (W) and the feature vector for the two right words (R). Each feature vector is used separately in order to train a multilayer perceptron (MLP) with a single hidden layer. Formally, the architecture is described by the following equations:

$$H_{1,X} = f(P_{1,X} \times X + b_{1,X}) \tag{1}$$

where X represents respectively the three feature vectors (L,W and R), P_i is the weight matrix and b_i is the bias vector.
The resulting vectors $H_{1,L}$, $H_{1,W}$ and $H_{1,R}$ are concatenated to form the first hidden layer $H1$. The H_1 vector is presented as the input of the second *MLP-MS* hidden layer H_2 computed according to the equation:

$$H_2 = g(P_2 \times H_1 + b_2) \tag{2}$$

Finally, the output layer is a vector O_k of k=2 nodes corresponding to the 2 labels $correct$ and $error$:

$$O_k = q(P_O \times H_2 + b_O) \tag{3}$$

Note that in our experiments f and g are respectively rectified linear units ($ReLU$) and hyperbolic tangent ($tanh$) activation functions, and q is the $softmax$ function.

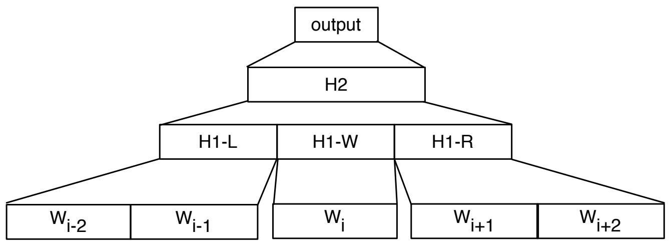

Figure 1. MLP-MS architecture for ASR error detection task.

2.2 Feature vectors

In this section, we describe the features collected for each word and how they are extracted. Some of these features are nearly the same as the ones presented in [1]. The word feature vector is the concatenation of the following features:

- **ASR confidence scores**: confidence scores are the posterior probabilities generated from the ASR system (PAP). The word posterior probability is computed over confusion networks, which is approximated by the sum of the posterior probabilities of all transitions through the word that are in competition with it.
- **Lexical features**: lexical features are derived from the word hypothesis output from the ASR system. They include the word length that represents the number of letters in the word, and three binary features indicating if the three 3-grams containing the current word have been seen in the training corpus of the ASR language model.
- **Syntactic features**: we obtain syntactic features by automatically assigning part-of-speech tags (POS tags), dependency labels – such label is a grammatical relation held between a governor (head) and a dependent –, and word governors, which are extracted from the word hypothesis output by using the MACAON NLP Tool chain[2] [16] to process the ASR outputs.
- **Linguistic word representation (embedding or symbol)**: The orthographic representation of a **word** is used in CRF approaches as for instance in [2]. Using our neural approach we can handle different word embeddings, which permits us to take advantage of the generalizations extracted during the construction of the continuous vectors.
- **Acoustic word embeddings**: these vectors represents the pronunciation of a word as a projection in a space with high dimension. Words projected into a close area are words acoustically similar [3].

[2] http://macaon.lif.univ-mrs.fr

2.3 Linguistic word embeddings: a combination-based approach

Different approaches have been proposed to create word embeddings through neural networks. These approaches can differ in the type of the architecture and the data used to train the model. In this study, we distinguish two categories of word embeddings: the ones estimated on unlabeled data, and others estimated on labeled data (dependency-based word embeddings). These representations are detailed respectively in the next subsections.

2.3.1 *Word embeddings based on unlabeled data*

This section presents three types of word embeddings coming from two available implementations (word2vec [15] and GloVe [19]):

- **Skip-gram**: This architecture from [15] takes as input the target word w_i and outputs the preceding and the following words.
  The target word W_i is at the input layer, and the context words C are at the output layer. It consists on predicting the contextual words C given the current word w_i.
  The skip-gram model with negative sampling seeks to represent each word W_i and each context C as d-dimensional vectors (V_{W_wi},V_C) in order to have similar vector representations for similar words. This is done by maximizing the dot product $V_{W_wi}.V_C$ associated with the good word-context pairs that occur in the document D and minimize it for negative examples, that do not necessarily exist in D. These negative examples are created by stochastically corrupting the pairs (W_i, C), thus the name *negative sampling*.
  Also, the context is not limited to the immediate context, and training instances can be created by skipping a constant number of words in its context, for instance, w_{i-3}, w_{i-4}, w_{i+3}, w_{i+4}, hence the name *skip-gram*.
- **GloVe**: This approach is introduced by [19], and relies on constructing a global co-occurrence matrix X of words, by processing the corpus using a sliding context window. Here, each element X_{ij} represents the number of times the word j appears in the context of word i.
  The model is based on the global co-occurrence matrix X instead of the actual corpus, thus the name GloVe, for Global Vectors.
  This model seeks to build vectors V_i and V_j that retain some useful information about how every pair of words i and j co-occur, such as:

  $$V_i^T V_j + b_i + b_j = log X_{ij} \tag{4}$$

  where b_i and b_j are the bias terms associated with words i and j, respectively.
  This is accomplished by minimizing a cost function J, which evaluates the sum of all squared errors:

  $$J = \sum\sum f(X_{ij})(V_i^T V_j + b_i + b_j - log X_{ij})^2 \tag{5}$$

  where f is weighting function which is used to prevent learning only from very common word pairs. The authors define the f as follows [19]:

  $$f(X_{ij}) = \begin{cases} \frac{X_{ij}}{X_{max}} & \textit{if } X_{ij} < X_{max} \\ 1 & \textit{otherwise} \end{cases}$$

2.3.2 *Dependency-based word embeddings*

Levy *et al.* [13] proposed an extension of word2vec, called word2vecf and denoted **w2vf-deps**, which allows to replace linear bag-of-words contexts with arbitrary features.

This model is a generalization of the skip-gram model with negative sampling introduced by [15], and it requires labeled data for training. As in [13], we derive contexts from dependency trees: a word is used to predict its governor and dependents, jointly with their dependency labels. This effectively allows for variable window size.

2.3.3 *Word embedding combination*

In the framework of this work, we have experimented different ways to combine the word embeddings presented above. Like described in a previous paper [10], the use of an auto encoder is very effective.

3 Acoustic word embeddings

3.1 Building acoustic word embeddings

The approach we used to build acoustic word embeddings is inspired from the one proposed in [3]. Word embeddings are trained through a deep neural architecture, depicted in figure 2, which relies on a convolutional neural network (CNN) classifier over words and on a deep neural network (DNN) trained by using a triplet ranking loss [3, 21, 22]. This architecture was proposed in [3] with the purpose to use the scores derived from the word classifier for lattice rescoring. The two architectures are trained using different inputs: speech signal and orthographic representation of the word.

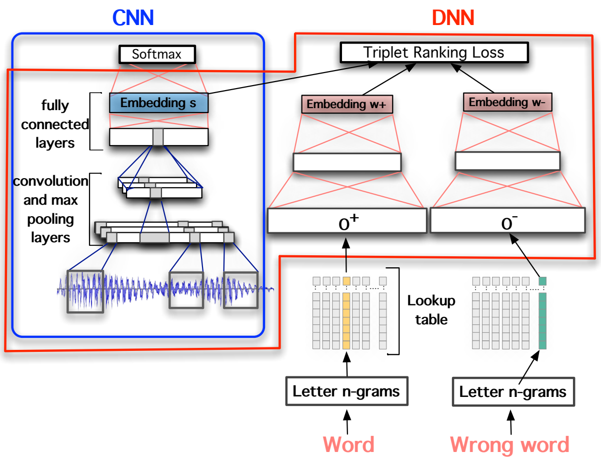

Figure 2. Deep architecture used to train acoustic word embeddings.

The CNN is trained to predict a word given an acoustic sequence of T frames as input. It is composed of a number of convolution and pooling layers, followed by a number of fully connected layers which feeds into the final softmax layer. The final fully connected layer just below the softmax one is called embedding layer **s** (it was called **e** in [3]). It contains a compact representation of the acoustic signal. This representation tends to preserve acoustic similarity between words, such that words are close in this space if they sound alike.

The idea behind using the second architecture is to be able to build an acoustic word embedding from orthographic word representation, especially in order to get an acoustic word embeddings for words not already observed in an audio speech signal. More, a such acoustic word embedding derived from an orthographic representation can be perceived as a canonical acoustic representation for a word, since different prononciations imply different embeddings **s**.

Like in [3], orthographic word representation consists on a bag of n-grams ($n \leq 3$) of letters, composed of 10222 trigrams, bigrams, and unigrams of letters, including special symbols *[* and *]* to specify the start and the end of a word. Then, we use an auto-encoder to reduce the size of this bag of n-grams vector to d-dimension. To check the performance of the resulting orthographic representation, a neural network is trained to predict a word given this orthographic representation. It reaches 99.99% of accuracy on the training set composed of $52k$ words of the vocabulary, showing the richness of this representation.

Similar to [3], a DNN was trained by using the triplet ranking loss [3, 21, 22] in order to project the orthographic word representation to the acoustic embeddings **s** obtained from the CNN architecture, which is trained independently. It takes as input a word orthographic representation and outputs an embedding vector of the same size as **s**. During the training process, this model takes as inputs the acoustic embedding **s** selected randomly from the training set, the orthographic representation of the matching word $\mathbf{o}^+$, and the orthographic representation of a randomly selected word different to the first word $\mathbf{o}^-$. These two orthographic representations supply shared parameters in the DNN.

We call $t = (\mathbf{s}, \mathbf{w}^+, \mathbf{w}^-)$ a triplet, where **s** is the acoustic signal embedding, $\mathbf{w}^+$ is the embedding obtained through the DNN for the matching word, while $\mathbf{w}^-$ is the embedding obtained for the wrong word. The triplet ranking loss is defined as:

$$Loss = \max(0, m - Sim_{dot}(s, w^+) + Sim_{dot}(s, w^-)) \quad (6)$$

where $Sim_{dot}(x, y)$ is the dot product function used to compute the similarity between two vectors x and y, and m is a margin parameter that regularizes the margin between the two pairs of similarity $Sim_{dot}(\mathbf{s}, \mathbf{w}^+)$ and $Sim_{dot}(\mathbf{s}, \mathbf{w}^-)$. This loss is weighted according to the rank in the CNN output of the word matching the audio signal.

The resulting trained model can then be used to build an acoustic embedding ($\mathbf{w}^+$) from any word, as long as one can extract an orthographic representation from it.

3.2 Experiments

3.2.1 Experimental data

Experimental data for ASR error detection is based on the entire official ETAPE corpus [11], composed by audio recordings of French broadcast news shows, with manual transcriptions (reference). This corpus is enriched with automatic transcriptions generated by the LIUM ASR system, which is a multi-pass system based on the CMU Sphinx decoder, using GMM/HMM acoustic models. This ASR system won the ETAPE evaluation campaign in 2012. A detailed description is presented in [4].

The automatic transcriptions have been aligned with reference transcriptions using the *sclite*[3] tool. From this alignment, each word in the corpora has been labeled as correct (C) or error (E). The description of the experimental data, in terms of size, word error rate (WER) as well as percentage of substitution (Sub), deletion (Del) and insertion (Ins), is reported in Table 1.

Name	**#words ref**	**#words hyp**	**WER**	**Sub**	**Del**	**Ins**
Train	349K	316K	25.3	10.3	12.0	3.1
Dev	54K	50K	24.6	10.3	11.0	3.3
Test	58K	53K	21.9	8.3	10.9	2.7

Table 1. Description of the experimental corpus.

The performance of the proposed approach is compared with a state-of-the-art system based on CRFs [2] provided by the *Wapiti* tagger[4] [12] and applied to the set of features presented in Section 2.2. The ASR error detection systems (MLP-MS and CRF) are trained on the training corpus (Train) and are applied on the test (Test) set. The development set (Dev) was used to tune all the parameters: the learning rate, the batch size and the hidden layers size of MLP-MS, and the features template of CRF, that describes which features are used in training and testing.

The performance is evaluated by using recall (R), precision (P) and F-measure (F) for the misrecognized word prediction and global Classification Error Rate (CER). CER is defined as the ratio of the number of misclassifications over the number of recognized words.

The linguistic word embedding described in Section 2.3 are made of 200 dimensions. They were computed from a large textual corpus, composed of about 2 billions of words. This corpus was built from articles of the French newspaper "Le Monde", the French Gigaword corpus, articles provided by Google News, and manual transcriptions of about 400 hours of French broadcast news.

The training set for the convolutional neural network used to compute acoustic word embedding consists of 488 hours of French Broadcast News with manual transcriptions. This dataset is composed of data coming from the ESTER1 [8], ESTER2 [9] and EPAC [6] corpora.

It contains $52k$ unique words that are seen at least twice each in the corpus. All of them corresponds to a total of 5.75 millions occurrences. In French language, many words have the same pronunciation without sharing the same spelling, and they can have different meanings; *e.g.* the sound [so] corresponds to four homophones: *sot* (fool), *saut* (jump), *sceau* (seal) and *seau* (bucket), and twice more by taking into account their plural forms that have the same pronunciation: *sots*, *sauts*, *sceaux*, and *seaux*. When a CNN is trained to predict a word given an acoustic sequence, these frequent homophones can introduce a bias to evaluate the recognition error. To avoid this, we merged all the homophones existing among the $52k$ unique words of the training corpus. As a result, we obtained a new reduced dictionary containing $45k$ words and classes of homophones.

Acoustic features provided to the CNN are log-filterbanks, computed every 10ms over a 25ms window yielding a 23-dimension vector for each frame. A forced alignment between manual transcriptions and speech signal was performed on the training set in order to detect word boundaries. The statistics computed from this alignment reveal that 99% of words are shorter than 1 second. Hence we decided to represent each word by 100 frames, thus, by a vector of 2300 dimensions. When words are shorter they are padded with zero equally on both ends, while longer words are cut equally on both ends.

3.2.2 Experimental results

Experimental results are summarized in Table 2. In terms of global classification error rate, the proposed neural approach outperforms

[3] http://www.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

[4] http://wapiti.limsi.fr

the CRF, especially by using a combination of embeddings. It yields 5.8% of CER reduction compared to CRF by using only linguistic word embedding. By using also acoustic word embeddings, the CER reduction reached 7.2%. One can also notice that the use of an auto encoder to combine word embeddings is really useful to capture complementarities of different single linguistic word embeddings.

Approach	**Word Represent.**	**Label error**			**Global**
		P	**R**	**F**	**CER**
CRF *(baseline)*	*discrete*	*67.69*	*54.74*	*60.53*	*8.56*
Neural with single ling. word embed.	w2vf-deps	71.90	50.98	59.66	8.26
	Skip-gram	**74.45**	46.75	57.44	8.30
	GloVe	72.16	46.97	56.90	8.53
Neural with ling. word embed. combination	w2vf-deps $\oplus$ Skip-gram $\oplus$ GloVe	69.66	57.89	63.23	8.07
Neural with ling. word embed. combination and acoustic word embed.	w2vf-deps $\oplus$ Skip-gram $\oplus$ GloVe $\mathbf{+ s + w^{+}}$	70.09	**58.92**	**64.02**	**7.94**

Table 2. Comparison of the use of different types of word embeddings in MLP-MS error detection system on Test corpus.

4 Conclusion

In this paper, we have investigated the use of a neural network to detect ASR error. Specifically, we proposed to effectively represent words through linguistic and acoustic word embeddings.

Experiments were made on automatic transcriptions generated by LIUM ASR system applied on the ETAPE corpus (French broadcast news). They show that the proposed neural architecture, using the acoustic word embeddings as additional features, outperforms state-of-the-art approach based on the use of Conditional Random Fields, with a reduction of the classification error rate of 7.2%.

5 Acknowledgements

This work was partially funded by the European Commission through the EUMSSI project, under the contract number 611057, in the framework of the FP7-ICT-2013-10 call and by the Région Pays de la Loire.

REFERENCES

[1] Frédric Béchet and Benoit Favre, 'Asr error segment localisation for spoken recovery strategy', *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference*, (2013).

[2] Frédric Béchet and Benoit Favre, 'Asr error segment localization for spoken recovery strategy', in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6837–6841, (May 2013).

[3] Samy Bengio and Georg Heigold, 'Word embeddings for speech recognition.', in *INTERSPEECH*, pp. 1053–1057, (2014).

[4] Paul Deléglise, Yannick Estève, Sylvain Meignier, and Teva Merlin, 'Improvements to the LIUM French ASR system based on CMU Sphinx: what helps to significantly reduce the word error rate?', in *Interspeech*, Brighton, UK, (September 2009).

[5] Richard Dufour, Géraldine Damnati, and Delphine Charlet. Automatic error region detection and characterization in lvcsr transcriptions of tv news shows, 2012. Acoustics, Speech and Signal Processing (ICASSP).

[6] Yannick Estève, Thierry Bazillon, Jean-Yves Antoine, Frédéric Béchet, and Jérôme Farinas, 'The EPAC Corpus: Manual and Automatic Annotations of Conversational Speech in French Broadcast News.', in *LREC*. Citeseer, (2010).

[7] Yannick Estève, Mohamed Bouallegue, Carole Lailler, Mohamed Morchid, Richard Dufour, Georges Linarès, Driss Matrouf, and Renato De Mori, 'Integration of word and semantic features for theme identification in telephone conversations', in *6th International Workshop on Spoken Dialog Systems (IWSDS 2015)*, (2015).

[8] Sylvain Galliano, Edouard Geoffrois, Djamel Mostefa, Khalid Choukri, Jean-François Bonastre, and Guillaume Gravier, 'The ESTER phase II evaluation campaign for the rich transcription of French Broadcast News.', in *Interspeech*, pp. 1149–1152, (2005).

[9] Sylvain Galliano, Guillaume Gravier, and Laura Chaubard, 'The ESTER 2 evaluation campaign for the rich transcription of French radio broadcasts.', in *Interspeech*, volume 9, pp. 2583–2586, (2009).

[10] Sahar Ghannay, Yannick Estève, Nathalie Camelin, Benoit Favre, Camille Dutrey, Fabian Santiago, and Martine Adda-Decker, 'Word embeddings for ASR error detection: combinations and evaluation.', in *LREC*, (2016).

[11] Guillaume Gravier, Gilles Adda, Niklas Paulsson, Matthieu Carr, Aude Giraudel, and Olivier Galibert, 'The ETAPE corpus for the evaluation of speech-based TV content processing in the French language', in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, (2012).

[12] Thomas Lavergne, Olivier Cappé, and François Yvon, 'Practical very large scale CRFs', in *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 504–513. Association for Computational Linguistics, (July 2010).

[13] Omer Levy and Yoav Goldberg, 'Dependencybased word embeddings', in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pp. 302–308, (2014).

[14] Julie Mauclair, Yannick Estève, Simon Petit-Renaud, and Paul Deléglise, 'Automatic detection of well recognized words in automatic speech transcription', in *Proceedings of the International Conference on Language Resources and Evaluation : LREC*, (2006).

[15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, 'Efficient estimation of word representations in vector space', (2013).

[16] Alexis Nasr, Frédéric Béchet, Jean-François Rey, Benoît Favre, and Joseph Le Roux, 'Macaon: An nlp tool suite for processing word lattices', in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Systems Demonstrations*, pp. 86–91. Association for Computational Linguistics, (2011).

[17] C. Parada, M. Dredze, D. Filimonov, and F. Jelinek, 'Contextual information improves oov detection in speech', *in North American chapter of thes Association for Computational Linguistics (NAACL)*, (2010).

[18] Thomas Pellegrini and Isabel Trancoso, 'Improving asr error detection with non-decoder based features.', *INTERSPEECH*, 1950–1953, (2010).

[19] Jeffrey Pennington, Richard Socher, and Christopher D Manning, 'Glove: Global vectors for word representation.', in *EMNLP*, volume 14, pp. 1532–1543, (2014).

[20] Yik-Cheung Tam, Yun Lei, Jing Zheng, and Wen Wang, 'Asr error detection using recurrent neural network language model and complementary asr', *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, 2312–2316, (2014).

[21] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu, 'Learning fine-grained image similarity with deep ranking', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1386–1393, (2014).

[22] Jason Weston, Samy Bengio, and Nicolas Usunier, 'Wsabie: Scaling up to large vocabulary image annotation', in *IJCAI*, volume 11, pp. 2764–2770, (2011).