LSTM LANGUAGE MODEL ADAPTATION WITH IMAGES AND TITLES FOR MULTIMEDIA AUTOMATIC SPEECH RECOGNITION

Yasufumi Moriya, Gareth. J. F. Jones

ADAPT Centre, School of Computing, Dublin City University, Dublin 9, Ireland

ABSTRACT

Transcription of multimedia data sources is often a challenging automatic speech recognition (ASR) task. The incorporation of visual features as additional contextual information as a means to improve ASR for this data has recently drawn attention from researchers. Our investigation extends existing ASR methods by using images and video titles to adapt a recurrent neural network (RNN) language model with a long-short term memory (LSTM) network. Our language model is tested on transcription of an existing corpus of instruction videos and on a new corpus consisting of lecture videos. Consistent reduction in perplexity by 5-10 is observed on both datasets. When the non-adapted model is combined with the image adaptation and video title adaptation models for n-best ASR hypotheses re-ranking, additionally the word error rate (WER) is decreased by around 0.5% on both datasets. By analysing the output word probabilities of the model, it is found that both image adaptation and video title adaptation give the model more confidence in the choice of contextually correct informative words

Index Terms— ASR, LSTM, multimodal language model adaptation

1. INTRODUCTION

The growth in archives of multimedia content is increasing demands for effective spoken content retrieval (SCR) systems. While searching multimedia archives using visual content has attracted much interest, the basis of many SCR systems is the use of transcripts of the spoken information stream generated using automatic speech recognition (ASR) [1]. Unsurprisingly, it has been found that errors in spoken transcripts, particularly semantic errors, can degrade the effectiveness of search systems [2]. The difficulty of building a robust ASR system for multimedia content has been demonstrated in previous work. For example, it has been found that the word error rate of speech recognition on *YouTube* videos can reach 40% [3]. In the MGB challenge 2015, the best performing system has a WER of 30-40%, when recognising more spontaneous speech in comedy and TV dramas [4]. WERs of this order are sufficient to greatly affect the behaviour of SCR systems. The primary goal of our work is to improve ASR accuracy from the perspective of SCR, ultimately focusing in particular on the accurate recognition of words in transcripts useful for effective SCR.

Our focus in this work is on the incorporation of information from non-speech channels into ASR. This builds on recent research which has shown success in grounding contextual information from non-speech channels into an ASR system for multimedia content. For example, Gupta et al. [5] extracted place and object features (referred to as visual features) from visual signals of instruction videos, and adapted both acoustic and language models with these features. Their latest work on the same dataset incorporates the same visual features into end-to-end connectionist temporal classification (CTC) and sequence-to-sequence models [6]. Similarly, Sun et al. [7] demonstrated that object features extracted from images can be useful for recognition of the spoken version of the Flickr8k image caption data.

The underlying principle of this work follows the line of research developed in these existing investigations, and seeks to extend the grounding contextual information for more effective ASR on multimedia data in which audio quality is typically less controlled and encompasses more diverse topics than is typical in other ASR settings. In our current study, we investigate whether metadata associated with multimedia content can provide an ASR system with information about concepts and activities of the content able to improve ASR accuracy. Specifically, video titles are embedded into fixed-length vectors and fed to a recurrent neural network (RNN) language model incorporating a long-short term memory (LSTM) network as additional contextual information [8, 9]. As well as video titles, visual adaptation of the LSTM language model is also performed for comparison. These language models are used to re-rank n-best ASR hypotheses. The adapted models are tested on the existing CMU "How-to" corpus, and on a new corpus, that is a collection of lecture videos from Udacity (https://eu.udacity.com/) [5]. Consistent gain in perplexity is observed with contextual adaptation of the language model. Combination of the non-adapted model with adapted models also leads to WER reduction by around 0.5% on both the datasets. By analysing the behaviour of the models, it was found that the contextually adapted models have more confidence in prediction of semantically meaningful words, but prediction accuracy of frequent words such as "a" and "in"

would appear to be negatively impacted by this adaptation.

The remainder of this paper is organised as follows: Section 2 introduces the methods used in our work, Section 3 describes our experimental setup, Section 4 gives results of our study, Section 5 provides analysis, and Section 6 concludes and gives directions for further work.

2. METHODOLOGY

In this section we introduce the methodologies for our investigations. We first introduce feature extraction, followed by adaptation of LSTM language models, and finally re-ranking of n-best ASR hypotheses.

2.1. Feature extraction

As outlined in the previous section, in this work we explore the use of two contextual features in ASR: visual information and video titles. We introduce these in more detail here.

Application of visual features in ASR has been explored in several existing studies [5, 6, 7]. In the visual adaptation setting, each utterance is accompanied by an image, or a video frame. A convolutional neural network (CNN) model extracts a fixed-length vector from the image or the video frame. In addition, the model can be fine-tuned using a collection of images of places labeled with the type of place (e.g., "tennis court", and "school"). In [5], a CNN model consisting of 5 convolutional layers followed by 3 fully-connected layers (known as AlexNet) [10] was employed to obtain a 1000 dimensional object probabilities from video frames. They further fine-tuned the CNN model with the MIT Places dataset [11] containing 2.5 million images associated with 205 scene categories to extract 205 dimensional place probabilities from video frames. These two adaptation vectors were fed to both the acoustic, and language model. They found that the adapted models reduced the WER on the "How to" dataset.

In our work, image embedding is extracted from video frames by the penultimate layer (fc7) of a pre-trained 19-layer convolutional neural network model from the visual geometory group of Oxford university (VGG19) [12]. The VGG19 model was pre-trained on the ImageNet dataset for the object recognition task [13]. We did not fine-tune the CNN model on a particular dataset, as the visual feature was merely used for comparison to the video title feature. We refer to the visual feature as *i-emb* throughout the paper.

The second contextual feature that we use is video titles. These serve as very short summaries of the multimedia content. We hypothesise that keywords present in video titles are good predictors of the domain of language spoken in the related video. At the pre-processing stage, the NLTK part-of-speech tagger identifies nouns, verbs, and adjectives among words in video titles [14]. The remainder of the words are removed to retain only content words, i.e. it is assumed that frequent function words such as "the" and "to" in video titles do not carry meaningful semantics associated with transcripts. A bag of words representation is then created by removing multiple occurrences of the same word. Unlike i-emb, several utterances can belong to one video, and one video title vector can be shared by multiple utterances.

The video title vector is produced by either the internal word embedding module of the language model, or the external fastText library [15]. fastText is an open-source library, that can transform a word into a continuous vector, even if the word is out-of-vocabulary, since the model is trained on word substrings. The potential advantage of using the internal word embedding module for creating video title representation is that the module can be tuned both for video title embedding and for general word embedding, both of which are input to the LSTM language model. The vectors can either be summed together or averaged to form a video title feature. It is known that summation and subtraction of word vectors can induce a new concept (e.g., king - man + woman = queen) [16]. On the other hand, averaging word vectors is claimed to be a good sentence representation according to [17, 18]. To explore the best approach to inducing video title representation for the language model adaptation, the video title vector is produced by (1) the internal word embedding module, and external fastText, and by (2) summation and averaging of word vectors. Comparative results of above approaches are provided in the results section. The video title feature is referred to as *t-emb* throughout the paper.

2.2. LSTM language model adaptation

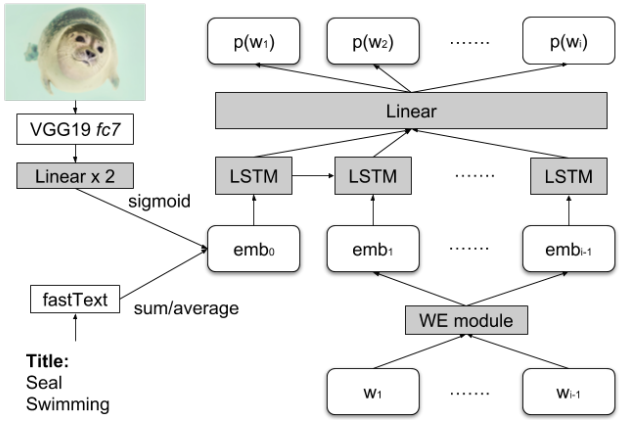


Fig. 1. Adaptation of an LSTM language model with either i-emb or t-emb. A raw image is transformed into i-emb by going through VGG19 up to the fc7 layer, two fully connected layers and a sigmoid function. Summing, or averaging of word embedding vectors creates t-emb. Weights of the components of the model coloured in grey are updated during training.

Figure 1 shows adaptation of the model performed in our work. The LSTM language model takes either i-emb or t-emb as initial input to predict the first word of a sentence w_1. The LSTM cell propagates this contextual information for prediction of the rest of the words in the sentence [9]. When i-emb is fed to the language model, it goes through two fully connected layers to match the dimensionality of i-emb with that of the word embedding input. A sigmoid function is set on top of the two fully connected layers. The weights of the two fully connected layers are updated during training of the language model. As mentioned above, t-emb is either the sum or average of the word embedding vectors of a video title. In Figure 1, fastText creates word embedding vectors of the title. The internal word embedding module (WE module in the figure) was also tested for this purpose, as mentioned in Section 2.1. The output of an LSTM layer is fed to one fully connected layer before the softmax function.

When image embedding is input, this architecture is similar to one of the first neural image caption generation models proposed by [19]. The difference between image caption generation and multimodal ASR, is that all the training examples of image caption data have strong correlation between text and images, whereas spoken transcripts of ASR data are not always strongly associated with an accompanying image or video title.

2.3. Re-ranking n-best ASR hypotheses

Both adapted and non-adapted models compute a negative log likelihood of a hypothesis as in Equation 1.

$$L_{RNN}(h) = -\sum_{i=1}^{I} logP(x_i|x_0...x_{i-1}) \quad (1)$$

where h is a hypothesis, I is the number of words in the hypothesis, and x_0 is i-emb or t-emb, when the model is adapted, and a start of sentence symbol <sos>, when the model is non-adapted. The total cost of the hypothesis is interpolated with the cost of the acoustic model as in Equation 2.

$$L(h) = AM(h) + \frac{1}{M}\sum_{m=1}^{M} LM_m(h) \quad (2)$$

where $AM(h)$ is the negative log likelihood of the hypothesis of an acoustic model, $LM_m(h)$ is the negative log likelihood of the hypothesis of a language model, and M is the number of language models to be interpolated. In this work, M is set to 2 when the LSTM language model is combined with the n-gram language model, and to 4 when the non-adapted, i-emb and t-emb models are all combined with the n-gram model. Since the likelihood is a negative log probability, the lower the score, the better the hypothesis.

3. EXPERIMENTAL SETUP

This section describes the two datasets used for our experiments, the procedures used to obtain the n-best ASR hypotheses of the recognised utterances for these datasets, and the configuration of the LSTM language model.

3.1. Datasets

As outlined above, two multimedia datasets were used for our ASR experiments.

3.1.1. Udacity Lecture Video Collection

The first is a collection of lecture videos from Udacity. Lecture videos show lecture slides and speakers. Transcripts of the videos were downloaded from each Udacity lecture website. All the numbers and symbols in transcripts were removed or expanded to words, and letters were converted to uppercase. Raw audio files and transcripts were preprocessed as follows:

- step 1: Audio files were aligned with transcripts through first-pass decoding with a deep neural network (DNN) acoustic model trained on the LibriSpeech corpus, and a biased language model. The language model was "biased", as the model was trained only on transcripts of a corresponding audio file. Decoded output was segmented into utterances at the point where silence and a sentence boundary coincide [20]. If an utterance was longer than 30 seconds, it was further split into sub-segments at the longest silence of the utterance.
- step 2: After second-pass decoding, the decoded output of utterances was compared to their references. All utterances, that had a mismatch with a reference were rejected since the transcripts may not be accurate.
- step 3: An internal DNN acoustic model was trained on clean utterances kept in the step 2. The decoding was performed on utterances segmented in step 1 with the internal DNN acoustic model, and with the biased language model from transcripts. This reduced the number of potential mismatches of speech with the transcripts, since the model was internal, and the quality of decoding output was improved.

The process to retain clean utterances of the corpus was based on the development of the LibriSpeech corpus [20]. In total, the collection contains 60 courses on engineering, science and programming topics, each with video title available. 50 courses were allocated for the training set, and 5 courses each for the validation and the test sets. Although some speakers appear in multiple courses, data was carefully partitioned, so that speakers in the training and validation sets

are not present in the test set. The duration of the training set is 163 hours, and that of the dev and test sets is roughly 3.5 hours each. The total vocabulary size of all 60 courses is 21,619.

3.1.2. CMU "How to" corpus

The other dataset used is the CMU "How-to" corpus consisting of a collection of instruction videos [5, 6]. Due to copyright issues, pre-processing of the corpus is different from that reported in [5, 6] and their results are not directly comparable to ours. We pre-processed the corpus in the same manner as the Udacity corpus. The full version of the dataset was used for experiments. The total duration of the corpus is 523 hours, which is roughly 3 times larger than the Udacity corpus. The same data partition was used as that described in [5, 6]. 512 hours of data are allocated for the training set, 5.5 hours for the validation set, and 4.7 hours for the test set. All 19,804 videos have a title. The vocabulary size of all of the data is 38,596.

3.2. Generation of n-best ASR hypotheses

The ASR system used to generate n-best hypotheses was built with Kaldi [21]. The acoustic model was trained with the nnet3 module, and the n-gram language model for decoding word graphs was 3-gram with modified Kneser-Ney interpolation using the SRILM toolkit [22, 23, 24]. The validation set in the experiments was only used to find the best epoch of the LSTM language model, and was not decoded by ASR. For this reason, the n-gram language model was trained on the training set combined with the validation set. N was set to 30 in the experiments, this value was also used in [5].

3.3. LSTM language model configuration

Both the non-adapted and contextually adapted language models were two-layer LSTMs, whose hidden size was set to 512. The video frames were transformed into 4096 dimensional vectors by VGG19. Then, the size of vectors was further reduced from 4096 to 512, and from 512 to 100 by two fully connected layers. The size of the LSTM hidden layers, and the intermediate size of image embedding were decided to be 512, which was empirically better than the size 256 and 1024.

For given input, the LSTM language model produced output probabilities of the next word. The probabilities were used to compute a cross entropy error of a reference word. The error was accumulated through a sentence, and the weights of the language model was updated with the backpropagation through time algorithm.

The model training began with the learning rate 20, and was divided by 4 when no improvement was seen on the validation set after each training epoch. The model was trained with 50 epochs. Dropout was applied to the word embedding module and the LSTM layers with rate 0.2. The mini-batch size was set to 100 for the Udacity corpus, and 90 for the CMU How-to corpus. The start/end of a sentence symbol <sos> and <eos> were added to each training example of the non-adapted model, while only <eos> was added to data of adapted language models. This is because either t-emb or i-emb takes a position of <sos> to predict the first word of a sentence. Video frames were taken from the middle of each utterance. When no words remained after pre-processing of video titles described in section 2.1, a zero vector of size 100 was substituted for t-emb for utterances without an accompanied video title. This happened for 28 video titles of the CMU How-to corpus, all of which belonged to the training set. Out-of-vocabulary words were not explicitly modelled. Implementation of the language models was based on PyTorch [25].

Table 1. Comparative results of different methods to produce t-emb on the Udacity and CMU How-to dataset. The internal word embedding module is "internal", and the fastText library is "fastText" [15]. "sum" and "ave" indicate summation and averaging of word embedding vectors. PPL is perplexity, and WER is word error rate.

	Udacity		How-to	
	PPL	WER	PPL	WER
internal, sum	138.50	14.67	63.21	18.33
internal, ave	138.87	14.70	64.12	18.44
fastText, sum	129.07	14.49	61.16	18.40
fastText, ave	131.69	14.48	60.03	18.27

4. RESULTS

Table 1 shows comparison of approaches to generating t-emb. Overall, employing an external library (i.e., fastText) to produce t-emb was better than the internal word embedding module. Both perplexity and WER were lower on both datasets, except summation of word embedding with fastText which slightly increased the WER compared to summation of word embedding with the internal module. There is not a large difference between summation and averaging of word embedding to form t-emb in these metrics. Thus, pre-computing t-emb before training a language model is our recommended approach.

Table 2 presents perplexity and WER of the non-adapted model, and the adapted models with i-emb and t-emb on the Udacity and CMU How-to corpus. Compared to the non-adapted model, i-emb reduced perplexity by 5 on both corpora, and t-emb by 5 on the Udacity corpus, and by more than 10 on the CMU How-to corpus. However, i-emb increased WER on the Udacity corpus by around 0.2%, and decreased WER on the How-to corpus by 0.12%. t-emb produced a comparative WER to the non-adapted model on the

Table 2. Experimental results for the non-adapted, i-emb, and t-emb models on the Udacity and CMU How-to datasets. t-emb is produced with fastText and averaging word embedding of video titles. "first-pass" is raw output of decoded transcripts without n-best hypothesis re-ranking. "oracle" is the best achievable WER of the n-best hypotheses. WERs of "combined" were obtained by averaging cost from the n-gram, non-adapted, i-emb and t-emb models, then interpolated with cost of the acoustic model.

	Udacity		How-to	
	PPL	WER	PPL	WER
first-pass	-	16.70	-	20.25
non-adapted	136.00	14.50	71.43	18.56
i-emb	131.29	14.71	65.79	18.43
t-emb	131.69	14.48	60.03	18.27
combined	-	13.97	-	17.98
oracle	-	9.20	-	15.13

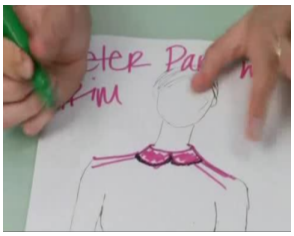
Udacity corpus, and reduced WER by 0.29% on the How-to corpus. When the non-adapted, i-emb and t-emb models were combined to re-rank hypotheses, WER was further reduced by 0.53% and 0.58% on the Udacity, and CMU How-to datasets, respectively.

Overall, both i-emb and t-emb models produced consistent improvement in perplexity on both the corpora. While lower perplexity resulted in lower WER on the CMU How-to dataset, this effect was not observed on the Udacity dataset. The image feature extractor was trained for the object recognition task. For this reason, i-emb could be more meaningful on the CMU How-to dataset, as videos in this corpus show objects such as a tennis racket and a basketball more often. In the Udacity corpus, the videos present lecture slides (e.g., characters, equations, flowcharts, and programming code), or humans (e.g., lecturers, and interviewees), which are not necessarily part of the ImageNet dataset. As for t-emb, video titles in the CMU How-to corpus tend to be longer (e.g., Intermediate Western Calligraphy Tips: The Acanthus Leaf & Calligraphy: Part 1) than in the Udacity corpus (e.g., How to Build a Startup). The language model could have more difficulty in learning with short video titles.

5. MODEL ANALYSIS

To further analyse how i-emb and t-emb affect prediction of words by the language model, one utterance each from the test split of the Udacity corpus and the CMU How-to corpus was fed to the non-adapted, i-emb and t-emb models. Embedding of video titles was produced with fastText and averaging of title word vectors. Softmax was used instead of log softmax to have more interpretable values.

Figure 2 shows word probabilities of a short region of an utterance taken from the test split of the CMU How-to corpus, its accompanied image, video title and the processed version of the video title. For this example, both i-emb and t-emb models gave a higher probability to "Pan", and "collar" than the non-adapted model. Interestingly, the accompanying image shows a collar-like object, and "Pan" and "collar" are included in the video title. Figure 3 shows the example utterance taken from the test split of the Udacity corpus. This utterance is taken from a Python introduction course. The image shows Python code, and the video title contains "Python". Both i-emb and t-emb models predicted "Python" with a higher probability than the non-adapted model. However, the non-adapted model had more confidence in prediction of common words such as "a" and "in" in these examples.



Transcript:
A Peter Pan collar is like ...
Video title:
Fashion Design for Small Collars: Fashion Design for Peter Pan Collar & Trim
Processed title:
Fashion Design Small Collars Peter Pan Collar Trim

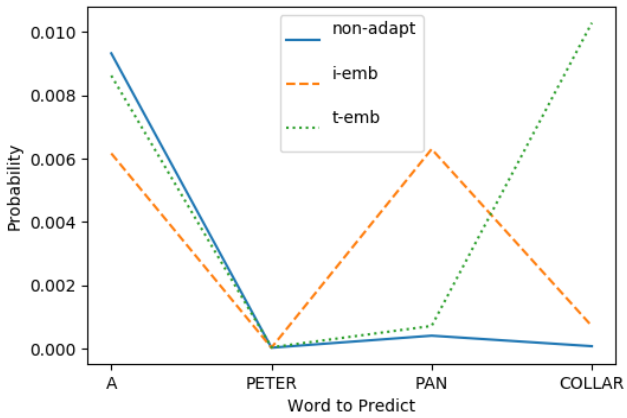
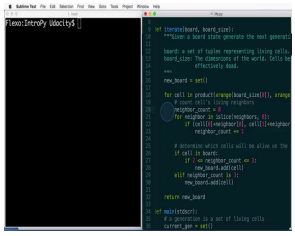


Fig. 2. One utterance from the test split of the How-to corpus with its aligned image, transcription, video title and the processed version of and video title. The lower graph shows probabilities of words produced by the adapted and non-adapted models.

Table 3 summarises averaged probabilities of nouns produced by the three models. Each noun is one of the keywords of a video. For example, most of the "testing" on the test set of the Udacity corpus occurs in the course "AB Testing for Analysis Business", and many examples of "nose" on the test set of the How-to corpus are found in "Cosmetics: Narrow the Nose with Makeup".

Generally, the model benefited from t-emb and i-emb for prediction of the words in the table. The t-emb model was especially good at giving higher confidence in the nouns listed



Transcript:
Indentation levels in Python are always ...
Video title:
Introduction to Python Programming
Processed title:
Introduction Python Programming

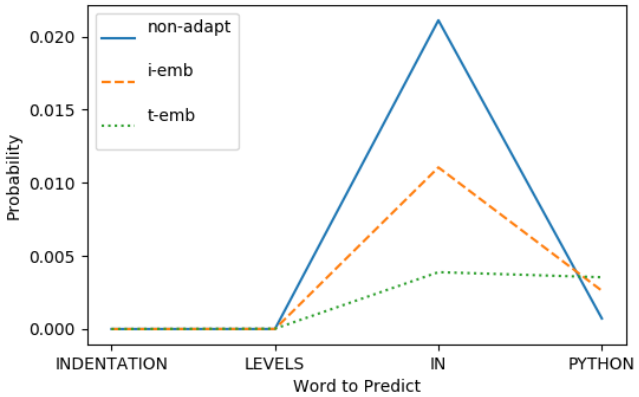


Fig. 3. One utterance from the test split of the Udacity corpus with its aligned image, transcription, video title and the processed version of and video title. The bottom graph shows probabilities of words produced by the adapted and non-adapted models.

in the table, except for "python" and "chord", in comparison to the non-adapted model. The i-emb model produced a higher or comparable probability for all but "python", "collar" and "golf" than the non-adapted model. This proved that the model was more aware of context, where speech was uttered, thanks to visual or video title information, so that semantically important words received a higher probability.

6. CONCLUSIONS AND FUTURE WORK

This work has investigated use of images and video titles for LSTM language model adaptation. Previous work showed that WER tends to be higher, when an ASR system is applied to multimedia data. This could be a bottleneck for SCR systems, as these systems rely on ASR transcripts for search operation on multimedia archives. It has also been demonstrated in previous work that adapting language and acoustic models with visual features can reduce WER on multimedia data. Our work extended previous methods by also adding video titles as contextual features, tested the adapted LSTM language model on existing and new multimedia corpora, and provided an in-depth analysis on behaviour of the models on keywords of the content.

Our findings are that: (1) when transforming a video title into a fixed-length vector, it is better to pre-train a word embedding module (e.g., fastText), and not to update the module weights jointly while training a language model; (2) a steady decrease in perplexity was obtained on both the Udacity and CMU How-to corpora with visual, and video title adaptation, while reduction in WER was only marginal; (3) the language model adapted with i-emb and t-emb tended to give a higher probability to keyword nouns of the content, while this might affect accuracy of prediction of common words such as "a" and "in".

Table 3. Word probabilities produced by the adapted and non-adapted models. The probabilities were averaged by the total occurrence on the test set. All of the probability values are multiplied by 100 for readability. The train column shows frequencies of the words on the training split of the corpora, and the test column on the test split of the corpora

	non-adapted	i-emb	t-emb	train	test
Udacity					
ad	0.06	0.04	0.37	107	205
analysis	0.08	0.34	0.19	263	47
python	0.44	0.19	0.41	467	43
readme	0.004	0.01	0.04	20	17
testing	0.18	1.33	1.73	367	12
How-to					
chord	21.56	20.86	20.82	495	10
collar	0.55	0.18	0.85	245	5
fish	2.34	6.07	14.49	812	36
golf	0.55	0.39	1.42	346	8
nose	0.6	5.01	6.85	752	11

For future work, several extensions of this work can be considered. Firstly, the model architecture in this work was quite basic, and a more sophisticated mechanism such as attention could be incorporated. Attention mechanisms have been actively investigated in the computer vision community for image caption generation [26, 27]. Nevertheless, such a model should be carefully designed, as the nature of data is different between image caption generation and multimedia ASR, as discussed in section 2.2. Secondly, rather than re-ranking n-best ASR hypotheses, re-scoring a lattice with an existing neural language model toolkit may be more effective for WER reduction [28]. Thirdly, contextual features could be further developed. For example, feeding multiple video frames for a single utterance to the language model instead of a single one image, or exploring more types of metadata.

7. ACKNOWLEDGEMENT

This work was supported by Science Foundation Ireland as part of the ADAPT Centre (Grant 13/RC/2106) (`www.adaptcentre.ie`) at Dublin City University.

8. REFERENCES

[1] M. Larson and G. J. F. Jones, "Spoken Content Retrieval: A survey of techniques and technologies," *Foundations and Trends in Information Retrieval*, vol. 4, no. 4-5, pp. 235–422, 2012.

[2] L. Lee, J. Glass, H. Lee, and C. Chan, "Spoken content retrieval: beyond cascading speech recognition with text retrieval," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1389–1420, 2015.

[3] H. Liao, E. McDermott, and A. Senior, "Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 368–373.

[4] P. Bell, M. J. F. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Wester, and P. C. Woodland, "The MGB challenge: Evaluating multi-genre broadcast media recognition," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 687–693.

[5] A. Gupta, Y. Miao, L. Neves, and F. Metze, "Visual features for context-aware speech recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5020–5024.

[6] S. Palaskar, R. Sanabria, and F. Metze, "End-to-End Multimodal Speech Recognition," *ArXiv 1804.09713*, 2018.

[7] F. Sun, D. Harwath, and J. Glass, "Look, listen, and decode: Multimodal speech recognition with images," in *Proceedings of IEEE Workshop on Spoken Language Technology (SLT)*, 2016, pp. 573–578.

[8] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of Interspeech*, 2010, pp. 1045–1048.

[9] S Hochreiter and J Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–80, 1997.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 1097–1105.

[11] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 487–495.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.

[13] Jia D., Wei D., R. Socher, Li-Jia L., Kai L., and Li F. F., "ImageNet: A large-scale hierarchical image database," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[14] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.

[15] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2016.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 3111–3119.

[17] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Towards Universal Paraphrastic Sentence Embeddings," *Proceedings of International Conference on Learning Representations (ICLR)*, 2016.

[18] S. Arora, Y. Liang, and T. Ma, "A simple but tough to beat baseline for sentence embeddings," *Proceedings of International Conference on Learning Representations (ICLR)*, 2017.

[19] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3156–3164.

[20] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011, pp. 1–4.

[22] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proceedings of Interspeech*, 2015, pp. 2–6.

[23] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, pp. 310–318, 1995.

[24] A. Stolcke, "SRILM-an extensible language modeling toolkit," in *Proceedings of International Conference on Spoken Language Processing (ICSLP)*, 2002.

[25] A. Paszke, G. Chanan, Z. Lin, S. Gross, E. Yang, L. Antiga, and Z. Devito, "Automatic differentiation in PyTorch," *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pp. 1–4, 2017.

[26] Ke. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–66, 2015.

[27] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing when to look: Adaptive attention via a visual sentinel for image captioning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3242–3250.

[28] X. Chen, X. Liu, Y. Qian, M. J. F. Gales, and P. C. Woodland, "CUED-RNNLM An open-source toolkit for efficient training and evaluation of recurrent neural network language models," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 6000–6004.