INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Semi-supervised Adaptation of RNNLMs by Fine-tuning with Domain-specific Auxiliary Features

Salil Deena[1], *Raymond W. M. Ng*[1], *Pranava Madhyastha*[2], *Lucia Specia*[2] *and Thomas Hain*[1]

[1]Speech and Hearing Research Group, The University of Sheffield, UK
[2]Natural Language Processing Research Group, The University of Sheffield, UK
{s.deena, wm.ng, p.madhyastha, l.specia, t.hain}@sheffield.ac.uk

Abstract

Recurrent neural network language models (RNNLMs) can be augmented with auxiliary features, which can provide an extra modality on top of the words. It has been found that RNNLMs perform best when trained on a large corpus of generic text and then fine-tuned on text corresponding to the sub-domain for which it is to be applied. However, in many cases the auxiliary features are available for the sub-domain text but not for the generic text. In such cases, semi-supervised techniques can be used to infer such features for the generic text data such that the RNNLM can be trained and then fine-tuned on the available in-domain data with corresponding auxiliary features.

In this paper, several novel approaches are investigated for dealing with the semi-supervised adaptation of RNNLMs with auxiliary features as input. These approaches include: using zero features during training to mask the weights of the feature sub-network; adding the feature sub-network only at the time of fine-tuning; deriving the features using a parametric model and; back-propagating to infer the features on the generic text. These approaches are investigated and results are reported both in terms of PPL and WER on a multi-genre broadcast ASR task.

Index Terms: RNNLM, Semi-supervised Adaptation, LDA topic models

1. Introduction

Language models (LMs) play a key role in automatic speech recognition (ASR) as they ensure that the output respects the pattern of the language in question. n-gram LMs dominated ASR for decades until RNNLMs [1] were introduced and found to give significant gains in performance. n-gram LM and RNNLM contributions are complementary and state-of-the-art ASR systems involve interpolation between the two types of models [1, 2, 3, 4, 5, 6, 7].

In automatic speech recognition, word context is generally heavily influenced by the domain, which can include topic, genre and speaking style. RNNLMs trained on a text corpus provide an implicit modelling of such contextual factors. It has been found that feature-based adaptation of RNNLMs by augmenting the input with domain-specific auxiliary features provide significant improvements in both perplexity (PPL) and word error rate (WER) [8, 2, 9, 10, 4, 6, 11]. Such features, however, can also include acoustic embeddings [12, 13] derived from audio, which might be available for only a subset of the text data, such as the matched in-domain data used for fine-tuning. In such cases, semi-supervised adaptation approaches can be used. The problem of features that only exist at fine-tuning but not at training time also exists in other areas where recurrent neural networks (RNN) are used, such as for Neural Machine Translation (NMT) [14]. The techniques proposed in this paper can be generalised to such cases.

In this work, the features used for domain adaptation of RNNLMs are latent Dirichlet allocation (LDA) [15] features extracted from the text. Whilst such features can be derived for the whole text, a setup is devised where the features are assumed to be available for the in-domain text but not for the generic text. Various methods are investigated for semi-supervised adaptation, aimed at compensating for the missing features and are evaluated against a setup having the complete set of features. The main contribution of this paper is showing that LDA features can be predicted for the generic text corpus, comprising about 80% of the whole data using the remaining 20%, giving a correlation of about 71% with ground truth LDA features.

2. Background and Related Work

2.1. Recurrent Neural Network LMs

Recurrent Neural Network Language Models [1] include a recurrent layer which can represent the full history $\mathbf{h}_i = < \mathbf{w}_{i-1}, \ldots, \mathbf{w}_1 >$ for word $\mathbf{w}_i$ using a concatenation of word $\mathbf{w}_{i-1}$ and the remaining context vector $\mathbf{v}_{i-2}$ from the previous time step. Each word $\mathbf{w}_i$ is represented using a 1-of-K encoding. An out-of-vocabulary (OOV) node [8, 9, 4] can be included at the input to represent any input word that is not in the chosen vocabulary, and an out-of-shortlist (OOS) node [8, 9, 4] can be included at the output to represent any word not in a shortlist vocabulary. The main purpose of the latter is to reduce the computational cost at the output layer by limiting the vocabulary to the most frequent words.

The LM probability for the next word $P(\mathbf{w}_{i+1}|\mathbf{w}_i, \mathbf{v}_{i-1})$ is computed as follows. A full history vector is obtained by concatenating $\mathbf{w}_i$ and the hidden (recurrent) layer activation from the previous time step, $\mathbf{v}_{i-1}$. The hidden layer takes the two inputs and produces a new representation of the history, $\mathbf{v}_i$ using a non-linear sigmoid activation. This activation is then input to the softmax activation function at the output layer to produce normalised RNNLM probabilities. Moreover, the activation from the hidden layer is also returned to the input layer, as it encodes the word history, and is used to compute the probability for the following word. A further auxiliary feature vector $\mathbf{f}$ can be provided as input to the network and this is illustrated in Figure 1. RNNLM training is performed using the back propagation through time (BPTT) algorithm [16], where the error is back-propagated through the recurrent connection for a specific number of time steps. The most expensive computation in RNNLM is the output softmax layer, which involves normalising the probabilities over the whole output vocabulary. In this paper, the approach proposed by Chen *et al.*[3, 17] is used, with GPU-based mini-batch training using spliced sentence bunch, allowing full softmax computation of the output using CE training.

http://dx.doi.org/10.21437/Interspeech.2017-1598

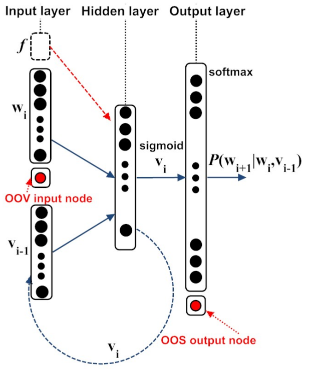

Figure 1: *Feature-based RNNLM with OOS and OOV nodes.*

2.2. Feature-based RNNLM Adaptation

In order to better understand at what level features can be added to the RNNLM, it is worth taking a look at the equations of the RNNLM to generate the current hidden state $\mathbf{v}_i$ from the previous hidden state vector and the current word, which is then used to predict the next word:

$$\mathbf{v}_i = RU(\mathbf{w}_i, \mathbf{v}_{i-1}) \tag{1}$$

$$= \sigma(\mathbf{W}^{(hh)}\mathbf{v}_{i-1} + \mathbf{W}^{(ih)}\mathbf{w}_i + \mathbf{b}^{(h)}) \tag{2}$$

$$\mathbf{w}_{i+1} \sim softmax(\mathbf{W}^{(ho)}\mathbf{v}_i + \mathbf{b}^{(o)}) \tag{3}$$

Where RU is the recurrent unit, σ is the sigmoid function and $\{\mathbf{W}, \mathbf{b}\}$ are the weights and biases of the neural network respectively.

There are four main approaches of adding features to a RNNLM [2, 18, 4, 19, 6]. Taking $\mathbf{f}$ to be the feature vectors, these approaches are:

2.2.1. Addition

In this particular method, the word vector $\mathbf{w}_i$ at each time step is added to the feature vector $\mathbf{f}$ and the result, $\mathbf{w}_i + \mathbf{f}$, is used as input to the recurrent unit, as follows:

$$\mathbf{v}_i = RU(\mathbf{w}_i + \mathbf{f}, \mathbf{v}_{i-1}) \tag{4}$$

This method, however, relies on the assumption that the feature vector and the word vector have the same dimensionalities, which can be also be enforced by projecting both vectors to a lower shared-dimensional subspace.

2.2.2. Stacking

Here, the word vector and the feature vector are concatenated such that the hidden state equation becomes:

$$\mathbf{v}_i = RU(\begin{bmatrix}\mathbf{w}_i \\ \mathbf{f}\end{bmatrix}, \mathbf{v}_{i-1}) \tag{5}$$

This approach was used in [18]. One disadvatange with this particular method is that the words and the features might correspond to different distributions. For example, the words are usually represented as one-hot vectors and the features can be continuous. Using the same weight matrix for both might lead to unexpected behaviour if the two distributions are very different.

2.2.3. Extra Perceptron

This involves using an extra perceptron which takes as input the current hidden state vector $\mathbf{v}_i$ together with a new weight matrix $\mathbf{W}^{(hh')}$ and the feature vector $\mathbf{f}$ with its corresponding weight matrix $\mathbf{W}^{(hf)}$, which outputs a custom hidden state vector $\mathbf{v}'_i$, that is then used to predict the next word.

$$\mathbf{v}_i = \sigma(\mathbf{W}^{(hh)}\mathbf{v}_{i-1} + \mathbf{W}^{(ih)}\mathbf{w}_i + \mathbf{b}^{(h)}) \tag{6}$$

$$\mathbf{v}'_i = \sigma(\mathbf{W}^{(hh')}\mathbf{v}_i + \mathbf{W}^{(hf)}\mathbf{f} + \mathbf{b}^{(h')}) \tag{7}$$

$$\mathbf{w}_{i+1} \sim softmax(\mathbf{W}^{(ho)}\mathbf{v}'_i + \mathbf{b}^{(o)}) \tag{8}$$

The advantage of this technique is that different learning rates can be set for the updates corresponding to equations 6, 7 and 8. However, the extra perceptron method requires more parameters to learn and can be prone to underfitting if the amount of data is small.

2.2.4. Feature Sub-network

In this particular method, used in [2, 4, 6, 20], a feature sub-network is added either at the hidden layer [2, 4, 6] or the output layer [20]. In line with previous work [6], we hereby consider using a separate weight matrix $\mathbf{W}^{(hf)}$ at the hidden layer. The hidden state vector equation then becomes:

$$\mathbf{v}_i = \sigma(\mathbf{W}^{(hh)}\mathbf{v}_{i-1} + \mathbf{W}^{(ih)}\mathbf{w}_i + \mathbf{W}^{(hf)}\mathbf{f} + \mathbf{b}^{(h)}) \tag{9}$$

This way of adding an extra feature seems to have the least side-effects and is the preferred method when applied to RNNLMs [2, 4, 6]. The reason is because it can cope with the word and feature vectors having different distributions and can also allow for control of over-fitting by setting different regularisation weights on $\mathbf{W}^{(hh)}$, $\mathbf{W}^{(ih)}$ and $\mathbf{W}^{(hf)}$.

In this work, the *feature sub-network* approach is used because it provides the most flexibility in terms of semi-supervised adaptation, as outlined in the next section. The *addition* method is definitely not a good choice because it assumes that both the words and features are available for all the data and that they are of the same dimensionality. The *stacking* approach is less flexible whilst the *extra perceptron* approach requires a lot more parameters than the *feature sub-network* method.

3. Semi-supervised Adaptation Methods

In this section, four novel approaches are proposed for dealing with RNNLM fine-tuning with auxiliary features, where such features are not available for the generic text used for training.

3.1. Zero Features to Mask the Feature Sub-network

Zero features allows the training of a RNNLM with the same structure as the final RNNLM where features are available whilst masking the feature sub-network and therefore give an equivalent network to one without features. The advantage of this approach is it does not require structural changes to the network at the time of fine-tuning, but it does require the whole network to reconfigure and recover the feature weight matrix $\mathbf{W}^{(hf)}$ during fine-tuning. It is expected that this reconfiguration would work well for a relatively small network but a large network would require a large amount of text data and epochs to converge.

3.2. Adding Feature Sub-network at Fine-tuning

This approach is similar in spirit to the zero features approach but can be convenient when an RNNLM has been trained already with possibly large amounts of data (and therefore requiring days to weeks to train) and then the network needs to be fine-tuned with in-domain text where features such as LDA are available. At the time of fine-tuning, the feature sub-network is introduced with the weights $\mathbf{W}^{(hf)}$ initialised with random values. During fine-tuning, the features are introduced and the weights of the feature sub-network are allowed to update and thus reach a new convergence point with the rest of the network parameters. The fact that structural changes need to be made to the network means that such an approach might not work very well in more complex networks where RNNs are used, such as Neural Machine Translation [14]. However, it is a good option for already trained models that would take too long to retrain.

3.3. Deriving the Features using Back-propagation

A back-propagation technique to infer document-level embeddings using an RNNLM was investigated in [20]. This involved training a RNNLM together with randomly initialised document-level features and then back-propagating till the features to recover them as document embeddings, whilst keeping the rest of the network fixed. A similar approach is used in this work, where the RNNLM is first trained with the show-level LDA features on the in-domain data as in [6]. The network is then kept fixed and back-propagation is carried out on the generic text in order to recover the features f for each sentence. The features updated at each time step and the feature vector obtained at the end of the sentence is taken to be the sentence-level feature. The sentence-level features are then averaged at the show-level, to give the target show-level LDA features, which are then used to train a feature-based RNNLM on the generic text. This method relies on the correlation between the text data and the features being captured on the in-domain text data prior to generalisation on the out-of-domain generic text.

3.4. Deriving the Features using a Parametric Model

In previous work [6], RNNLM adaptation was applied to Multi-Genre broadcast data [21], where genre labels are available for in-domain BBC show data but not for historical subtitles that were provided for language model training. In order to address this issue when training the RNNLM using both the large out-of-domain and the smaller in-domain data, the genre labels for the historical subtitles were derived using a parametric model. This involved extracting LDA features from both the in-domain and out-of-domain data and using a support vector machine (SVM) classifier to predict the genre labels from the LDA features, on the out-of-domain text, after training the SVM on the in-domain data.

A similar parametric approach can be used to predict any other type of features. In this work, a parametric regression model is proposed, which can learn a mapping between the hidden state vector of a separate RNNLM trained without features, and the feature vectors. The hidden state vectors are first generated at sentence level by feeding each sentence through the RNNLM and extracting the hidden states at the end of the sentence. These hidden state vectors are averaged across all sentences in a given show, in order to give show-level hidden state vectors. A parametric regression model is then fitted between the averaged hidden state vectors and the show-level LDA features according to Eqn 10.

$$\mathbf{f}_{sh} = g(mean_{s \in sh}(\mathbf{v}_s)) \qquad (10)$$

Where $\mathbf{f}_{sh}$ are features at the show level, g is the parametric function, $mean_{s \in sh}(\mathbf{v}_s)$ is the mean of sentence-level (s) hidden state vectors, $\mathbf{v}_s$, averaged for each show (sh).

Two parametric models are used in this work: linear regression and multi-layer perceptron (MLP) regression with 1024 nodes, which are then used to predict LDA features for the out-of-domain text.

4. Experiments and Results

4.1. Multi-Genre Broadcast Challenge Data

The experiments in this paper make use of the data provided by the British Broadcasting Corporation (BBC) for the Multi-Genre Broadcast (MGB) challenge 2015 [21]. Task 1 of the challenge involved participants having to perform the automatic transcription of a set of BBC shows. These shows were chosen to cover the multiple genres in broadcast TV, categorised in terms of 8 genres: advice, children's, comedy, competition, documentary, drama, events and news. Acoustic Model (AM) training data was fixed and limited to more than 2,000 shows, broadcast by the BBC during 6 weeks in April and May of 2008. The development data for the task consisted of 47 shows that were broadcast by the BBC during a week in mid-May 2008. The numbers of shows and the associated broadcast time for training and development data are shown in Table 1.

Table 1: *Amount of Training and Development Data.*

	Train		Development	
Genre	Shows	Time	Shows	Time
Advice	264	193.1h.	4	3.0h.
Children's	415	168.6h.	8	3.0h.
Comedy	148	74.0h.	6	3.2h.
Competition	270	186.3h.	6	3.3h.
Documentary	285	214.2h.	9	6.8h.
Drama	145	107.9h.	4	2.7h.
Events	179	282.0h.	5	4.3h.
News	487	354.4h.	5	2.0h.
Total	2,193	1580.5h.	47	28.3h.

Additional data was available for Language Model (LM) training in the form of subtitles from about $340k$ shows broadcast from 1979 to March 2008, with a total of 650 million words, and referred to as $LM1$, corresponds to the generic out-of-domain text. The subtitles from the 2,000+ shows for acoustic modelling with a total of $10.6M$ words, referred here as $LM2$, corresponds to our matched in-domain text.

The development data was used as the evaluation set in order to provide fair comparison with previous work [4, 22, 6]. For language model experiments, the $LM2$ data was partitioned into a training and development set by selecting 90% of text for each programme for training and the remaining 10% for development, after shuffling the lines for each programme.

4.2. Experimental Setup

The experimental setup is aimed at both evaluating the quality of the predicted LDA features against ground truth, and then testing the semi-supervised approaches presented in this paper in an ASR setup. The setup for ASR experiments is the same as in [23, 6] with a baseline 4-gram language model built on $LM1+LM2$ text by first selecting a vocabulary of 200k words was chosen from all the words in the $LM2$ text (87k) and augmented with the most frequently occuring words in $LM1$. The RNNLMs were trained with a $60k$ vocabulary for the input word list and a $50k$ vocabulary for the output word list, both obtained by shortlisting the $200k$ vocabulary based on most frequent words. ASR decoding was performed in three stages; in a first stage, lattices were generated using a 2-gram LMs, followed by lattice rescoring with a 4-gram LM to generate new lattices. n-best list rescoring was performed by first converting the lattices to n-best lists, with n being 100 as in [6].

The RNNLMs with 512 hidden nodes, are first trained on out-of-domain selected $LM1$ text and then fine-tuned on the in-domain $LM2$ text data. The out-of-domain data is generated by selecting $5M$ lines from the $LM1$ text, after randomly shuffling the shows, giving a total of about $45M$ words, corresponding to 11843 shows. This was done because training a RNNLM on the whole $LM1$ text takes more than a week and a representative subset of the data makes the experiments more manageable in a limited time-frame. The domain-specific features are derived by training a show-based latent Dirichlet allocation (LDA) topic model on the in-domain text data, as reported in previous work [6], and computing posteriors on both the in-domain and out-of-domain text data.

4.3. Experiments

4.3.1. Predicted LDA Features

The generated LDA features are compared against the ground truth by computing Pearson's correlation coefficient between the two. The Average Correlation Coefficient (ACC) is computed by averaging the coefficients across the 47 shows in the development set. The predicted LDA features are also used to predict the show they correspond to by computing the nearest neighbour (NN) to the ground truth LDA vectors for each show. It is to be noted that the shows are non-overlapping between the training and test sets, which makes this similar to the zero-shot learning problem [24], where LDA features are used to predict the shows to which they correspond to whilst such shows have not been seen before. The ACC results and the NN classification rate are given in Table 2.

Table 2: *Quality of Predicted LDA Features.*

Method	ACC	Show Classification
Parametric Model (Linear regr.)	0.5997 ± 0.1382	68.1%
Parametric Model (MLP regr.)	0.7144 ± 0.1360	78.7%
Back-Propagation	0.6289 ± 0.1523	69.0%

The results show that the MLP regression method outperforms both linear regression and the back-propagation method, giving the highest correlation to ground truth and also the highest show classification rate. This shows that the RNNLM hidden state vectors averaged at show level and the show-level LDA feature vectors are highly correlated, despite the fact that they are derived differently. The back-propagation method performs less well, possibly due to the vanishing gradient effect that is known to happen with RNNs [25].

4.3.2. ASR Experiments

Table 3 shows the perplexity on our 10% selected development text from $LM2$ both when training the generic RNNLMs and when fine-tuning on the in-domain data. The PPL are reported at the 1st and 10th epochs both for training and fine-tuning but a similar pattern is observed for further epochs of fine-tuning. In order to demonstrate the effect of the LDA features, random features of the same dimensionality as the 100-dim LDA features, that sum to 1, are generated for each show and used as features for the RNNLM at training. LDA features are then used at fine-tuning for all the methods.

Table 3: *RNNLM Training and Fine-tuning PPL.*

Method	Training		Fine-tuning	
	Epoch 1	Epoch 10	Epoch 1	Epoch 10
No Feat.	336.8	229.4	160.3	151.5
LDA Feat.	603.7	209.7	151.3	127.0
Random Feat.	461.4	231.8	163.9	146.6
Semi-supervised Adaptation Methods				
Zero Feat.	336.9	229.4	157.0	132.4
Add Feat. Sub-net.	336.9	229.4	149.0	128.8
LDA Feat. from Back-prop.	401.9	221.4	153.1	131.9
LDA Feat. from MLP regr.	380.3	205.8	148.5	127.1

The results show that LDA features lead to a big drop in perplexity, compared to random features of the same dimension. MLP regression method to predict LDA features lead to a final PPL that almost matches using original LDA features. The other semi-supervised approaches also lead to encouraging results with techniques that try to predict LDA features outperforming those that do not utilise or mask the features.

The baseline ASR results, using a 4-gram LM, and results obtained using interpolation RNNLMs (with optimal interpolation weight of 0.4 for the RNNLM) are given in terms of global PPL and WER in Table 4. The LDA features are extracted from the text output of first-pass decoding using the 4-gram LM.

Table 4: *ASR Results on MGB data.*

Genre	Adv.	Child.	Comed.	Compet.	Doc.	Dram.	Even.	News	Global	
System	WER								PPL	WER
4-gram	24.6	30.4	43.5	25.8	28.0	41.5	34.1	15.7	100.1	30.1
+RNNLM	24.0	29.6	43.6	25.3	27.6	42.5	33.1	14.8	94.7	29.7
+LDA feat.	23.7	29.1	43.3	24.8	27.2	41.3	32.6	14.7	86.6	29.2
Semi-supervised Adaptation Methods										
Zero Feat.	23.8	29.2	43.4	25.1	27.4	41.7	32.8	14.7	89.2	29.5
Add Sub-net.	23.7	29.2	43.3	25.1	27.3	41.6	32.8	14.7	89.3	29.4
Back-prop.	23.7	29.1	43.5	25.0	27.4	41.7	33.0	14.7	88.1	29.4
MLP regr.	23.5	29.2	43.3	25.0	27.3	41.2	32.7	14.7	86.7	29.3

These results confirm the PPL results in Table 3 and show that using an MLP regressor to predict LDA features at a show-level, leads to a background RNNLM trained with the predicted features, to match the performance of using ground-truth features, when fine-tuned on matched in-domain data. Other approaches such as using zero features, adding a feature sub-network at fine-tuning and using the back-propagation method, also lead to better results than not using features at all, but are outperformed by the MLP regression approach.

5. Conclusion

This paper deals with the semi-supervised adaptation of RNNLMs with domain-specific LDA features, when such features are available for matched in-domain text used for fine-tuning, but not for text data used for training the background language model. Four approaches are proposed to deal with the missing features, which include: using zero features to mask the feature sub-network; adding the feature sub-network at fine-tuning only; deriving the missing features using a back-propagation approach and deriving the missing features using a parametric model. It was shown that using multi-layer perceptron regression between RNNLM hidden state vectors averaged at a show level and show-level LDA features, to predict LDA features on out-of-domain text, gives the best results.

Future work will involve investigating better approaches to derive show-based RNNLM hidden states than averaging, such as the application of attention [14] as in the case of neural machine translation (NMT). The extension of the proposed approaches to the NMT domain will also be investigated, where additional modalities (such as audio and images) are only available for the in-domain text used for model fine-tuning.

6. Acknowledgements and Data Access

The audio and subtitle data used for the experiments was distributed as part of the MGB Challenge (mgb-challenge.org) through a licence with the BBC. The CTM and scoring files can be accessed via DOI: 10.15131/shef.data.4772890. This work was partly supported by the EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology) and the MultiMT H2020 ERC Starting Grant No. 678017.

7. References

[1] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH'10: Proc. of the 11th Annual Conference of the International Speech Communication Association*, 2010, pp. 1045–1048.

[2] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model." in *SLT'12: Proc. of the IEEE workshop on Spoken Language Technologies*, 2012, pp. 234–239.

[3] X. Chen, Y. Wang, X. Liu, M. J. Gales, and P. C. Woodland, "Efficient GPU-based training of recurrent neural network language models using spliced sentence bunch." in *INTERSPEECH'14: Proc. of the 11th Annual Conference of the International Speech Communication Association*, 2014, pp. 641–645.

[4] X. Chen, T. Tan, X. Liu, P. Lanchantin, M. Wan, M. J. F. Gales, and P. C. Woodland, "Recurrent neural network language model adaptation for multi-genre broadcast speech recognition," in *INTERSPEECH'15: Proc. of the 16th Annual Conference of the International Speech Communication Association*, 2015, pp. 3511–3515.

[5] W. Williams, N. Prasad, D. Mrva, T. Ash, and T. Robinson, "Scaling recurrent neural network language models," in *ICASSP'15: Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 5391–5395.

[6] S. Deena, M. Hasan, M. Doulaty, O. Saz, and T. Hain, "Combining Feature and Model-Based Adaptation of RNNLMs for Multi-Genre Broadcast Speech Recognition," in *Interspeech'2016: Proc. of the 17th Annual Conference of the International Speech Communication Association*, 2016, pp. 2343–2347.

[7] S. Reddy, P. Swietojanski, P. Bell, and S. Renals, "Unsupervised adaptation of recurrent neural network language models," in *Interspeech 2016*, 2016, pp. 2333–2337.

[8] J. Park, X. Liu, M. J. F. Gales, and P. C. Woodland, "Improved neural network based language modelling and adaptation," in *INTERSPEECH'10: Proc. of the 11th Annual Conference of the International Speech Communication Association*, 2010, pp. 1041–1044.

[9] T. Alumäe, "Multi-domain neural network language model," in *INTERSPEECH'13, 14th Annual Conference of the International Speech Communication Association*, 2013, pp. 2182–2186.

[10] O. Tilk and T. Alumäe, "Multi-domain recurrent neural network language model for medical speech recognition," in *Human Language Technologies - The Baltic Perspective - Proceedings of the Sixth International Conference Baltic HLT 2014, Kaunas, Lithuania, September 26-27, 2014*, 2014, pp. 149–152.

[11] M. A. Haidar and M. Kurimo, "LDA-Based Context Dependent Recurrent Neural Network Language Model Using Document-based Topic Distribution of Words," in *ICASSP'17: Proc. of the 42nd IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

[12] M. Doulaty, O. Saz, R. W. M. Ng, and T. Hain, "Latent dirichlet allocation based organisation of broadcast media archives for deep neural network adaptation," in *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015.

[13] ——, "Automatic genre and show identification of broadcast media," in *Interspeech'16: Proc. of the 17th Annual Conference of the International Speech Communication Association*, 2016, pp. 2115–2119.

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR'15: Proc. of the International Conference on Learning Representations*, 2015.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. Oct, pp. 533–536+, 1986.

[17] 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 *ICASSP'16: Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 6000–6004.

[18] Y. Shi, P. Wiggers, and C. M. Jonker, "Towards recurrent neural networks language models with linguistic and contextual features." in *INTERSPEECH'12: Proc. of the 13th Annual Conference of the International Speech Communication Association*. ISCA, 2012, pp. 1664–1667.

[19] C. D. V. Hoang, T. Cohn, and G. Haffari, "Incorporating side information into recurrent neural network language models," in *NAACL-HLT'16: Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1250–1255.

[20] A. Giel and R. Diaz, "Document embeddings via recurrent language models," University of Standford, Tech. Rep., 2016.

[21] P. Bell, M. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Webster, and P. Woodland, "The MGB challenge: Evaluating multi–genre broadcast media transcription," in *ASRU'15: Proc. of IEEE workshop on Automatic Speech Recognition and Understanding*, 2015.

[22] P. C. Woodland, X. Liu, Y. Qian, C. Zhang, M. J. F. Gales, P. Karanasou, P. Lanchantin, and L. Wang, "Cambridge university transcription systems for the multi-genre broadcast challenge," in *ASRU'15: Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 639–646.

[23] O. Saz, M. Doulaty, S. Deena, R. Milner, R. W. M. Ng, M. Hasan, Y. Liu, and T. Hain, "The 2015 Sheffield system for transcription of multi–genre broadcast media," in *ASRU'15: Proc. of the IEEE Automatic Speech Recognition and Understanding workshop*, 2015.

[24] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *NIPS'09: Advances in Neural Information Processing Systems*, 2009, pp. 1410–1418.

[25] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *ICML'13: Proc. of the 30th International Conference on Machine Learning*, 2013, pp. 1310–1318.