A UNIFIED FRAMEWORK FOR FEATURE-BASED DOMAIN ADAPTATION OF NEURAL NETWORK LANGUAGE MODELS

Michael Hentschel[⋆†], *Marc Delcroix*[†], *Atsunori Ogawa*[†], *Tomoharu Iwata*[†], *Tomohiro Nakatani*[†]

[⋆]Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, Japan
michael.hentschel.mc5@is.naist.jp
[†]NTT Communication Science Laboratories, NTT Corporation, 2-4, Hikaridai, Seika-cho, Kyoto, Japan
{marc.delcroix, ogawa.atsunori, iwata.tomoharu, nakatani.tomohiro}@lab.ntt.co.jp

ABSTRACT

An important task for language models is the adaptation of general-domain models to specific target domains. For neural network-based language models, feature-based domain adaptation has been a popular method in previous research. Conventional methods use an adaptation feature providing context information that is calculated from a topic model. However, such a topic model needs to be trained separately from the language model. To unify the language and context model training, we present an approach that combines an extractor network and a domain adaptation layer. The extractor network learns a context representation from a fixed-size window of past words and provides the context information for the adaptation layer. The benefit of our method is that the extractor network can be trained jointly with the language model in a single training step. Our proposed method showed superior performance over conventional domain adaptation with topic features on a dataset of TED talks with respect to perplexity and word error rate after 100-best rescoring.

***Index Terms*—** Domain adaptation, Topic model, Sequence summary network, Recurrent neural network language model

1. INTRODUCTION

Domain adaptation of language models (LM) has been a task that has seen some major interest in recent years. LMs are trained on general-domain data but are usually applied to specific domains during evaluation, which created the necessity for this research field. The task of domain adaptation has already been studied with count-based LMs [1, 2]. More recently, for neural network LMs (NN-LMs) two paradigms have evolved. First, model-based adaptation that adapts network weight by retraining with in-domain data [3, 4, 5]. Second, feature-based adaptation that uses an adaptation feature during training and evaluation to provide domain information [6, 7, 8]. As adaptation features, topic features from a latent Dirichlet allocation [9] (LDA) topic model have been commonly used. As previous research showed, these features are successful at providing some global topic information to the network. Feature-based domain adaptation showed reductions in perplexity (PPL) and word error rate (WER) in N-best rescoring, when applied to automatic speech recognition (ASR).

Training LMs with feature-based domain adaptation requires several steps. First, an LDA topic model has to be trained independently from the LM. This requires text pre-processing and the segmentation of the training data into documents. However, the training data might not always have this segmentation. Second, training the LDA topic model and extracting the features follows a very different scheme. The LDA features are calculated from a sliding window over the input text. In addition, the LDA features are not optimised for LM adaptation.

Facing this scenario, it is desirable to have a more integrated approach for feature-based domain adaptation. This is our motivation for UniFA, a Unified framework for Feature-based domain Adaptation. UniFA is a combined approach for training the context representation and the language model jointly in a single training step. It does not require any text pre-processing and the training data can be used in the same form for training the LM and the context representation. The model learns to extract the context features itself to improve word prediction.

To obtain a context representation in our framework, we use a sequence summary network [10] (SSN). The SSN learns to extract a context representation from a fixed-size window of past words. This context representation is used in an adaptation layer to calculate the adaptation parameters. We inserted an adaptation layer in the LM before the output layer. In UniFA, this adaptation layer is realised by feature-based learning hidden unit contributions [11, 12, 13] (fLHUC). In comparison with conventional LDA feature-based adaptation, the advantage of using such approach is that the context representation and the LM can be trained jointly by standard error back-propagation in a single training step. In contrast to an LDA topic model, our approach does not require any pre-processing of the text data. The SSN and the LM can both be trained using the same data.

Our experimental results on a dataset of TED talks showed that our proposed approach outperforms LDA feature-based domain adaptation in terms of PPL and we achieve competitive WER results in 100-best rescoring on the TED-LIUM dataset [14]. In addition to analysing the model performance, we also give an insight into the context representation and adaptation parameters learned by the extractor network. As we show in the experimental result section, there are similarities among the adaptation parameters learned by each method but our proposed method is able to capture more dynamic and local context.

2. LSTM-LM

NN-LMs based on recurrent networks are most common nowadays and they showed to significantly outperform count-based LMs [15, 16]. In a recurrent neural network LM (RNN-LM), the input word ID is encoded by a one-hot vector $\boldsymbol{w}(t)$. The input to the recurrent layer $\boldsymbol{x}(t)$ is calculated using the word embedding matrix $\boldsymbol{U}^{(\mathrm{w})}$

$$\boldsymbol{x}(t) = \boldsymbol{U}^{(\mathrm{w})}\boldsymbol{w}(t). \tag{1}$$

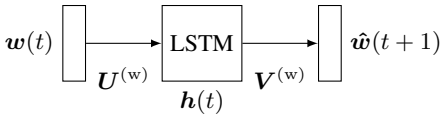

Fig. 1: A conventional LSTM-LM.

We use long short-term memory (LSTM) [17] units as recurrent units in all our RNN-LMs as shown in Figure 1. The LSTM outputs $\boldsymbol{h}(t)$ and keeps its state $\boldsymbol{c}(t)$. The recurrent layer is followed by a linear layer and the softmax function to calculate the probability for the next word $\hat{\boldsymbol{w}}(t+1)$

$$\hat{\boldsymbol{w}}(t+1) = \mathrm{softmax}(\boldsymbol{V}^{(\mathrm{w})}\boldsymbol{h}(t) + \boldsymbol{b}^{(\mathrm{V,w})}), \tag{2}$$

where $\boldsymbol{V}^{(\mathrm{w})}$ and $\boldsymbol{b}^{(\mathrm{V,w})}$ are the weight matrix and the bias vector of the output layer, respectively.

3. PROPOSED UniFA ADAPTATION FRAMEWORK

Our proposed adaptation framework UniFA shown in Figure 2 consists of two main parts, which we present in this section in more detail:

1. A context extractor network, that learns a fixed-length context representation from a window of past words
2. An adaptation layer, where the feature extractor network's output is used to adapt the LSTM cells' output

3.1. Context Extractor Network

For our context representation, we use a context extractor network based on a sequence summary network [10] (SSN) shown in Figure 2 (a). SSNs have been more common in speaker adaptation for acoustic models and this is the first time to apply this technique for LM adaptation. There are usually other more common methods for context representations in natural language processing (see Section 5), but we decided to use the SSN because it is computationally very efficient. We only have to compute the output of a (shallow) feed-forward network, which can easily be done in parallel for the whole context window on a GPU.

The SSN takes as input a context window of word embeddings $[\boldsymbol{x}(t), \boldsymbol{x}(t-1), \ldots, \boldsymbol{x}(t-N-1)]$ where N is the size of the context window. The context window covers the current word and the previous $N-1$ words. Our experiments showed that it is beneficial to use shared word embeddings for the SSN and the language model. In this way, we can share the parameters for the embedding matrix and reduce the total number of parameters. The SSN consists of M feed-forward (FFWD) layers followed by a non-linearity. We used rectified linear units (reLU) as non-linearity. The SSN then computes outputs $[\boldsymbol{y}(t), \boldsymbol{y}(t-1), \ldots, \boldsymbol{y}(t-N-1)]$ for each element in the window

$$\boldsymbol{y}(t-n) = \mathrm{FFWD}(\boldsymbol{x}(t-n)), \quad \forall n \in \{0, 1, \ldots, N-1\}. \tag{3}$$

The outputs of the SSN are averaged over the whole context window to obtain a fixed-size vector representation of the whole sequence

$$\boldsymbol{a}(t) = \frac{1}{N}\sum_{n=0}^{N} \boldsymbol{y}(t-n). \tag{4}$$

The context representation $\boldsymbol{a}(t)$ is used as adaptation feature for the LM.

In [10], it was shown that such an SSN could be trained jointly with the network it is attached to by error backpropagation for speaker adaptation. That means it fulfils our requirement that the context representation is jointly trainable with the main network.

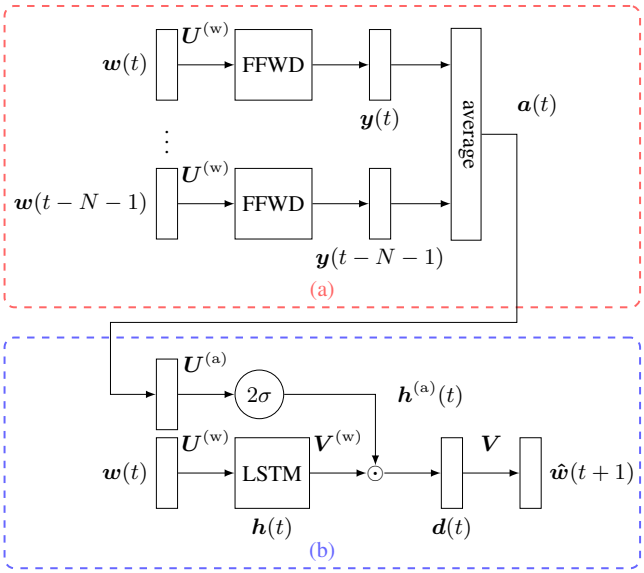

Fig. 2: UniFA adaptation framework with (a) the sequence summary network (SSN) based context extractor network, and (b) LSTM-LM domain adaptation with fLHUC.

3.2. fLHUC Domain Adaptation Layer

The context adaptation is realised in our proposed approach by feature-based learning hidden unit contributions (fLHUC). We insert an adaptation layer after the LSTM before the output layer as shown in Figure 2 (b). The context representation $\boldsymbol{a}(t)$ calculated by the context extractor network is used as adaptation feature for the fLHUC. fLHUC applies a gating function to the output of the LSTM

$$\boldsymbol{d}(t) = (\boldsymbol{V}^{(\mathrm{w})}\boldsymbol{h}(t) + \boldsymbol{b}^{(\mathrm{V,w})}) \odot \boldsymbol{h}^{(\mathrm{a})}(t), \tag{5}$$

where $\odot$ denotes an element-wise multiplication of two vectors. The adaptation parameters $\boldsymbol{h}^{(\mathrm{a})}(t)$ are calculated from the context features by a linear layer and a subsequent sigmoid activation function

$$\boldsymbol{h}^{(\mathrm{a})}(t) = 2\sigma(\boldsymbol{U}^{(\mathrm{a})}\boldsymbol{a}(t) + \boldsymbol{b}^{(\mathrm{U,a})}). \tag{6}$$

$\boldsymbol{U}^{(\mathrm{a})}$ and $\boldsymbol{b}^{(\mathrm{U,a})}$ are the weight matrix and bias vector of the linear layer for the output of the SSN. This linear layer is necessary to match the output dimensionality of the context extractor network and the adaptation layer.

In fLHUC, the range of the adaptation parameters $\boldsymbol{h}^{(\mathrm{a})}(t)$ is restricted to $[0, 2]$. The amplification of the sigmoid function by a factor of two compensates for the reduced activation in some nodes after the adaptation layer. To keep the activation of the nodes after the adaptation layer on approximately the same level [11] suggested amplifying the activation of those nodes that are not set to zero.

When combining the output of the SSN with the adaptation layer, we found it very helpful to use a normalisation of the context features. We applied layer normalisation [18] to the input of the sigmoid function.

4. EXPERIMENTS

4.1. Dataset

The dataset for our experiments consisted of TED talks. For LM training, we used a training set consisting of subtitles from 2494 TED talks. The validation and test sets were composed of subtitles

Table 1: Comparison of subtitle and TED-LIUM test sets.

	sentence length (words)		
	min	max	mean
subtitle	2	19	9
TED-LIUM	2	122	25

as well. The training set had approximately 5.1M tokens and a vocabulary size of 43K words, where every word appearing only once is mapped to the OOV token.

For ASR experiments, we used a speech recogniser based on the standard TED-LIUM Kaldi recipe [14, 19], i.e., a speech recogniser with a feed-forward deep neural network acoustic model without any sequence discriminative training. The validation and test sets in the TED-LIUM recipe and in the subtitle-based set contained the same talks, but TED-LIUM uses re-transcribed talks. This re-transcription introduced some mismatch with the text data that we used for training our LMs. The major difference is the sentence length in the subtitle and TED-LIUM test sets as shown in Table 1. By re-transcribing the talks, the sentence length in TED-LIUM increased compared with the original subtitles.

4.2. Model Training

For all NN-LMs in the experiments, we used state of the art LSTMs as recurrent units. The recurrent layer had 300 LSTM cells. We used AdaGrad [20] optimiser and a starting learning rate of 0.1. The Gradients were clipped to an L2 norm of 5. We used standard back-propagation through time for 20 time steps and a mini-batch size of 128. The networks were regularised by dropout [21] with a dropout ratio of 50%. As mentioned in Section 3.2, we used layer normalisation [18] when combining the SSN with the fLHUC adaptation layer. All methods were implemented with the open-source toolkit chainer [22]. The number of model parameters of the baseline LSTM-LM were 26M. Using fLHUC, the parameter size increased by 105K. UniFA increased the parameter size of the baseline by 270K.

In the experiments we compared fLHUC with two different methods to derive the adaptation features ($\boldsymbol{a}(t)$ in Figure 2 (b)). The first method was conventional LDA feature-based domain adaptation (fLHUC-LDA). We used the same fLHUC adaptation layer [13] as in our proposed method but the adaptation parameters were derived from LDA features. In this case, we used the LDA implementation in Scikit-learn [23] to train the topic model and to calculate the features. We set the number of LDA topics to 50. The topic model was trained by splitting the subtitle training set into individual talks. For training and evaluation of the LMs, the LDA features were calculated from a fixed-size sliding window. The second method uses the SSN to extract the context features. The SSN in our proposed UniFA had 300 units and we used a network with a single hidden layer.

For N-best rescoring, we used the 100-best list to calculate the LDA features and to calculate the context features with our proposed method. That means, recognition errors in the hypothesis can have an effect on the context representation of subsequent utterances if an utterance is shorter than the context window.

4.3. PPL Results

We first compare the PPL results as summarised in Table 2. All PPLs for NN-LMs were obtained without N-gram interpolation and show therefore a fair comparison of the different adaptation mechanisms. As baseline, we provide the PPL of an LSTM-LM without any domain adaptation.

As comparison to our proposed method, we used conventional LDA feature-based domain adaptation, with LDA features calculated from a window size of 50, 100 and 200 words, respectively. From the adaptation features, we calculated the adaptation parameters in fLHUC. fLHUC-LDA showed slight improvements on the LSTM-LM for the subtitle set and TED-LIUM for longer context window lengths. On TED-LIUM, a window size of 200 words led to a 23% and 12% PPL reduction for the validation and test set compared with a 50-word window size, respectively.

Table 2: PPL for subtitle and TED-LIUM validation and test set. The number in brackets denotes the context window size.

Model	Subtitle PPL		TED-LIUM PPL	
	val	test	val	test
LSTM-LM	51.58	51.98	209.34	156.29
fLHUC-LDA (50)	48.32	48.56	226.48	154.15
fLHUC-LDA (100)	47.47	47.44	188.33	139.12
fLHUC-LDA (200)	46.76	46.98	173.51	135.68
UniFA (50)	**35.74**	**36.82**	**144.09**	**120.14**
UniFA (100)	38.27	37.66	165.55	129.21
UniFA (200)	37.18	37.82	168.79	135.34

Table 3: WER after 100-best rescoring for TED-LIUM. The number in brackets is the context window size.

Model	val WER[%]	test WER[%]
1-best	16.3	15.1
LSTM-LM	14.2	12.1
fLHUC-LDA (50)	14.3	12.2
fLHUC-LDA (100)	14.0	12.2
fLHUC-LDA (200)	14.0	11.9
UniFA (50)	13.8	**11.8**
UniFA (100)	**13.7**	12.0
UniFA (200)	13.9	12.0

For our proposed method, we used the same window sizes as with fLHUC-LDA. Compared with all other methods, our proposed method achieved a significantly lower PPL on the subtitle set and TED-LIUM. The PPL reduction ranges from 26% to 19% on the subtitle set. Especially for small context window sizes, our method showed a high PPL reduction compared with fLHUC-LDA. On TED-LIUM, our proposed method consistently outperformed our LSTM-LM baseline and could further improve on fLHUC-LDA.

4.4. Rescoring Results

Another very important metric for NN-LMs used in ASR systems is their improvement of the recognition result after rescoring in terms of WER. For this purpose, we used Kaldi's TED-LIUM recipe for 100-best rescoring. The 1-best result was obtained with a trigram LM that is distributed with the TED-LIUM recipe [24]. The trigram was trained on different data than TED talks. Table 3 shows the WER for the 1-best decoding result and after 100-best rescoring for all models. The baseline LSTM-LM led to a great WER reduction compared with the 1-best result. For fLHUC-LDA , the ASR results were analogue to the PPL results. With a short context window, fLHUC-LDA fell behind the LSTM-LM. The method could only improve on the baseline with a larger context window.

Our proposed UniFA showed again an improvement on an LSTM-LM across all context window sizes. The validation set WER was in all cases better than for fLHUC-LDA and it only fell behind fLHUC-LDA for the longest window size of 200 words. A matched-pair significance test showed a significant improvement of UniFA (50) on an LSTM-LM at a significance level $p < 5\%$.

As Table 1 shows, for TED-LIUM the average utterance length is 25 words. This corresponds exactly to half the window length of our best UniFA model. LMs do not benefit much from a very large context window in rescoring because there might be many recognition errors from preceding utterances in the context window. However, these errors seem not to harm the model at shorter context windows and the extractor network can successfully provide additional (short-term topic) information.

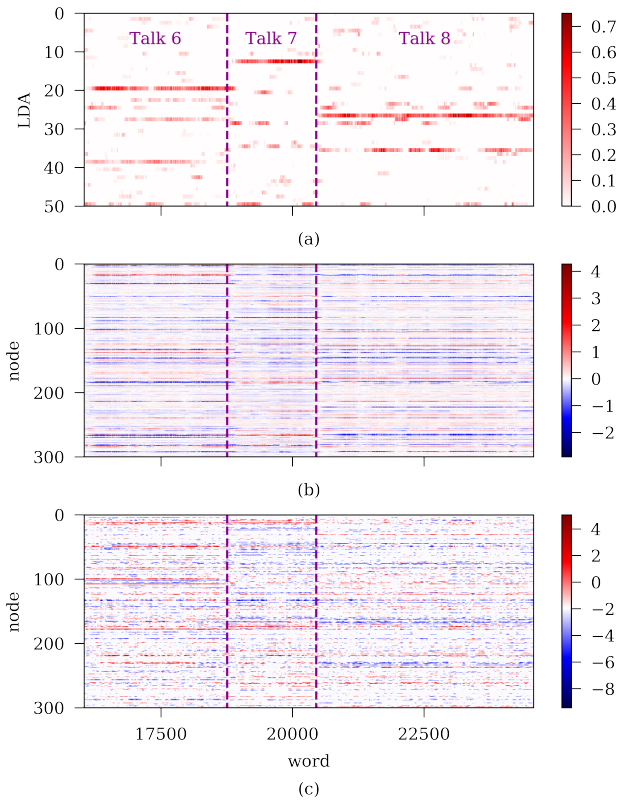

Fig. 3: Visualisation of (a) LDA features for 200-word window size, fLHUC adaptation parameters before the sigmoid function from (b) LDA features from 200-word window size and (c) SSN (50) for talks six, seven and eight in the subtitle test set.

4.5. Analysis of fLHUC Adaptation Parameters

In addition to comparing each model's effectiveness in ASR, we provide further investigation of the adaptation parameters learned by each model. We compare the best models with LDA features and SSN from Section 4.4. Figure 3 shows a visualisation for the fLHUC adaptation parameters before the sigmoid function when learned from LDA features and from the SSN with different window lengths. As data for the comparison, we used talks six, seven and eight in the subtitle test set. Figure 3 (a) and (b) show the LDA features for a 200-word window and the adaptation parameters the network learned to extracted from them, respectively. The LDA features show three distinct topics with high activation for each of the talks. The adaptation parameters have values around zero for most of the nodes in the adaptation layer, which means that the node passes through its input. However, certain nodes show either high (red horizontal lines) or low (blue horizontal lines) activation depending on the active LDA topic. This means that these nodes amplify or attenuate their input, respectively.

Figure 3 (c) shows the adaptation parameters when learned to extract with the SSN from a context window of 50 words[1]. Similar to the adaptation parameters learned from LDA features, there are also some nodes in the adaptation layer that receive constant high or low activations during each talk. This suggests that the SSN learns to capture a global topic-like context spanning several thousand words. However, the adaptation parameters from the SSN appear noisier compared with the ones learned from LDA features. This means that the SSN output changes more frequently depending on the context compared with LDA features. This suggests that in addition to the long term context the SSN can also capture more local context. This leads to a more frequent regulation of each node in the adaptation layer. As the experimental results showed, LDA features with shorter context windows were unable to capture these local topic changes but it was important for reducing PPL and WER.

[1]We applied a threshold to values around minus two for improved visualisation.

5. RELATED WORK

We decided to use an SSN for the context representation, however, there are other more common methods in natural language processing. Among these are convolutional neural networks [25, 26], or vector based representations like paragraph vector [27] and derivatives thereof [28]. Despite being a successfull method, paragraph vector is not suitable to our problem because it requires a document matrix that grows with the number of documents. A row in this matrix serves as representation for each document and the matrix has to be extended for unknown documents in the evaluation set. Another popular and successful context representation is the encoder-decoder framework [29], which showed to be very successful in machine translation [30] or in the generation of conversation responses [31] among other tasks. However for our application, such encoder architecture would be computationally very expensive because we have to run an LSTM-based encoder over a long (possibly a few hundred words) context window at each and every word prediction.

A recent approach for LM domain adaptation which is related to ours was presented in [32]. It uses a mixture of pre-trained LSTM-LMs where the weight for each LSTM-LM is determined by a mixer network. This mixer network is represented by another LSTM. However, there are several differences with our proposed approach. First, the method introduced in [32] is not a pure feature-based domain adaptation method because it requires domain information for the pre-training. Second, it is a multi-step training process whereas our UniFA is a single-step training process.

6. SUMMARY AND OUTLOOK

We presented a unified framework UniFA for feature-based LM domain adaptation based on a sequence summary network (SSN) and feature based learning hidden unit contributions (fLHUC). Our results on a dataset of TED talks showed improved PPL results compared with a baseline LSTM-LM and conventional feature-based domain adaptation with LDA features. In 100-best rescoring on the TED-LIUM dataset, our proposed method UniFA consistently improved on an LSTM-LM baseline and outperformed conventional feature-based adaptation in all but one cases. In addition, we provided further insight into what the network learns from the context window by analysis of the fLHUC adaptation parameters.

For future work, we are considering to improve on the current context representation learned by the SSN. So far, longer context window sizes were not helpful further improving the results. We are planning to investigate pre-training the SSN for instance as an auto-encoder. In addition, we think that attention could help to obtain a better context summary vector. For longer context windows, performance might degrade because the context vector only contains an average representation of all words in the context window. Especially regarding rescoring, we consider learning a more robust context representation by introducing distortions in the training data. An interesting approach regarding how these distortions could be generated was presented in [33].

7. REFERENCES

[1] Ronald Rosenfeld, "Two decades of statistical language modeling: Where do we go from here?," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1270–1278, 2000.

[2] Jerome R Bellegarda, "Statistical language model adaptation: review and perspectives," *Speech communication*, vol. 42, no. 1, pp. 93–108, 2004.

[3] Junho Park, Xunying Liu, Mark JF Gales, and Phil C Woodland, "Improved neural network based language modelling and adaptation," in *INTERSPEECH*, 2010.

[4] Ottokar Tilk and Tanel Alumäe, "Multi-domain recurrent neural network language model for medical speech recognition.," in *Baltic HLT*, 2014, pp. 149–152.

[5] Tanel Alumäe, "Multi-domain neural network language model.," in *INTERSPEECH*, 2013, vol. 13, pp. 2182–2186.

[6] Tomas Mikolov and Geoffrey Zweig, "Context dependent recurrent neural network language model.," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2012, vol. 12, pp. 234–239.

[7] Daniel Soutner and Luděk Müller, "Application of LSTM neural networks in language modelling," in *International Conference on Text, Speech and Dialogue*. Springer, 2013, pp. 105–112.

[8] Xie Chen, Tian Tan, Xunying Liu, Pierre Lanchantin, Moquan Wan, Mark JF Gales, and Philip C Woodland, "Recurrent neural network language model adaptation for multi-genre broadcast speech recognition," in *INTERSPEECH*, 2015.

[9] David M Blei, Andrew Y Ng, and Michael I Jordan, "Latent Dirichlet allocation," *Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[10] Karel Veselỳ, Shinji Watanabe, Katerina Žmolíková, Martin Karafiát, Lukáš Burget, and Jan Honza Černockỳ, "Sequence summarizing neural network for speaker adaptation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5315–5319.

[11] Pawel Swietojanski and Steve Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 171–176.

[12] Lahiru Samarakoon and Khe Chai Sim, "Subspace LHUC for fast adaptation of deep neural network acoustic models.," in *INTERSPEECH*, 2016, pp. 1593–1597.

[13] Michael Hentschel, Marc Delcroix, Atsunori Ogawa, and Tomohiro Nakatani, "Feature Based Learning Hidden Unit Contributions for Domain Adaptation of RNN-LMs," in *2018 Asia-Pacific Signal and Infromation Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018.

[14] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, "TED-LIUM: an automatic speech recognition dedicated corpus.," in *LREC*, 2012, pp. 125–129.

[15] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH*, 2010, pp. 1045–1048.

[16] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney, "LSTM neural networks for language modeling.," in *INTERSPEECH*, 2012, pp. 194–197.

[17] Sepp Hochreiter and Jürgen Schmidhuber, "Long Short-Term Memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[19] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. Dec. 2011, IEEE.

[20] John Duchi, Elad Hazan, and Yoram Singer, "Adative subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[21] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[22] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton, "Chainer: a next-generation open source framework for deep learning," in *Workshop on Machine Learning Systems (LearningSys) in the Twenty-ninth Anual Conference on Neural Information Processing (NIPS)*, 2015.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[24] Will Williams, Niranjani Prasad, David Mrva, Tom Ash, and Tony Robinson, "Scaling recurrent neural network language models," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015*. IEEE, 2015, pp. 5391–5395.

[25] Yoon Kim, "Convolutional neural networks for sentence classification," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1746—1751.

[26] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014, vol. 1, pp. 655–665.

[27] Quoc Le and Tomas Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning (ICML)*, 2014, pp. 1188–1196.

[28] Minmin Chen, "Efficient vector representation for documents through corruption," in *International Conference on Learning Representations (ICLR)*, 2017.

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2014, pp. 3104–3112.

[30] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[31] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan, "A neural network approach to context-sensitive generation of conversational responses," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 196–205.

[32] Kazuki Irie, Shankar Kumar, Michael Nirschl, and Hank Liao, "RADMM: recurrent adaptive mixture model with applications to domain robust language modeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6079–6083.

[33] Yinghui Huang, Abhinav Sethy, Kartik Audhkhasi, and Bhuvana Ramabhadran, "Whole sentence neural language model," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6089–6093.