Recurrent Neural Network Based Personalized Language Modeling by Social Network Crowdsourcing

Tsung-Hsien Wen[1]*,Aaron Heidel*[1]*, Hung-Yi Lee*[1]*, Yu Tsao*[2]*, and Lin-Shan Lee*[1.2]

[1]National Taiwan University,
[2]Academic Sinica, Taipei, Taiwan
r00921033@ntu.edu.tw, lslee@gate.sinica.edu.tw

Abstract

Voice interface on smartphones has been accepted and used by many people nowadays, which boosts the need for a personalized language model to better model the linguistic patterns and wording habits of that particular smartphone owner. Thanks to the popularity of social networks in recent years, personal texts and messages are no more unaccessible. However, data sparseness is still a unsolved problem. In this paper, we propose a three-step adaptation approach to personalize recurrent neural network language models (RNNLMs). We believe its capability of modeling word histories as distributed representation and arbitrary length can help to mitigate the data sparseness problem. Furthermore, additional user oriented features were also proposed to empower RNNLMs stronger capabilities for personalization. The experiments on Facebook dataset showed that the proposed method not only drastically reduced the model perplexity in preliminaries, but also moderately reduced the word error rate in n-best rescoring tests.

Index Terms: Recurrent Neural Network, Personalized Language Modeling, Social Network, LM adaptation

1. Introduction

Personalization, which is an indispensable aspect in several real world applications nowadays, has been studied in a wide range of research fields, such as personalized web search [1, 2] and recommendation systems [3, 4, 5, 6]. In speech community, acoustic model adaptation [7, 8, 9], which has been proven to give an impressive word error rate reduction, can be viewed as the major efforts made for personalization in this field. However, although acoustic model personalization has been well studied, there's little works on personalized language modeling. Traditionally, language model adaptation [10, 11, 12] stressed on the problem of cross-domain linguistic mismatch in which cross-individual linguistic mismatch is often ignored. Probably because of the lack of large enough personal corpora in early days, it was hard to realize the idea, and therefore we have to aggregate the corpora produced by many different individuals but on similar domains to perform domain-oriented language model adaptation. But now, there're two major trends that strongly push the progress of personalized language models: First, thanks to the mass production and rapid proliferation of smartphones in recent years, everyone can have one, which makes the use of smartphones a very personal experience [13, 14, 15]. As a consequence, only a speaker dependent recognizer is needed. Secondly, large quantities of texts are available over the social networks with known authors and given relations among the authors. It is possible to train personalized language models because it may be reasonable to assume that users with close relationships may share common subject topics, wording habits and linguistic patterns.

N-gram-based language models [10, 16, 17] including various adaptation techniques have been proven to work very well for many applications. In our previous work, an N-gram-based language model was adapted toward one particular user's wording patterns by incorporating social texts the target user and other users had posted considering different aspects of similarities between users [18]. However, the lack of natural strategy to model long range dependencies [19, 20, 21] and the potentially error-prone back-off behaviors [22] seem to limit the prediction power of N-gram models. Recently, several studies had shown that the neural network-based (NN) language models [23, 24, 25] improved over N-gram-based models by taking the advantage of the learned distributed representations of word histories. Moreover, RNNLMs [26, 27, 28], by memorizing arbitrary length of histories in a recurrent structure, modeled long range dependencies in an elegant way. Considering our language model personalization task, we think it is beneficial to adopt RNNLMs due to the following three reasons: First, the amount of social posts we can obtain from each individual are still relatively sparse compared to the amount that we can robustly apply N-gram model adaptation, but the distributed representation of word histories can extenuate this problem by addressing the similarities of combinatorial number of originally disjoint sentences in the continuous space. Secondly, online messages or social posts tend to be relatively casual thus usually not strongly obey traditional grammar rules in which short-term dependencies may not be evident enough for predicting the next word anymore. The recurrent structure may help to mitigate it since it memorizes more dependencies than N-gram does. Last but not least, besides posts and messages, the social network also maintains a lot of information such as user profiles, relationships, interactions, tastes, and interests. These are all valuable information we can use while predicting one's linguistic habits. As mentioned in some previous works [28, 29], adding additional auxiliary features into RNNLMs is relatively easy and intuitive. As a result, we value the meet of LM personalization task and RNNLMs as an indispensable attempt.

In this paper, we propose a new paradigm for personalized language modeling in which we use RNNLMs as our base model. The system is built on a voice access of cloud application [18]. A crowdsourcing mechanism is designed to collecting texts and other information on social networks from users. A three-step training for personalized RNNLMs was also proposed considering the feasibility and complexity of the training process. Some user oriented features were incorporated into the model for better word prediction capabilities. The experiments showed that the proposed method not only drastically reduced the model perplexity in preliminaries, but also moderately reduced the word error rate in n-best rescoring tests.

2. The Crowdsourcing Platform

In order to obtain personal acoustic and language data, we had implemented a cloud-based application in which it can help user to access his social network via voice, which was shown

in Fig. 1. As a consequence, this cloud-based application can also be viewed as a crowdsourcing platform for collecting personal data. The idea of crowdsourcing [30, 31] had been applied for several purposes in many different fields. For example, a crowdsourcing approach [32] was proposed to collect queries in an information retrieval system considering temporal information. MIT movie browser [33, 34] also relied on Amazon Mechanical Turk, the most famous crowdsourcing platform, to build a crowd-supervised spoken language system. The definition of crowdsourcing varied, in our case, an implicit crowdsourcing [31] is at play: the user login his Facebook account for our voice access service by granting our application the authority to collect his acoustic and linguistic data for adaptation. At the same time, the user could enjoy the benefits of better accuracy brought by the personalized recognizer that we had built from those crawled data.

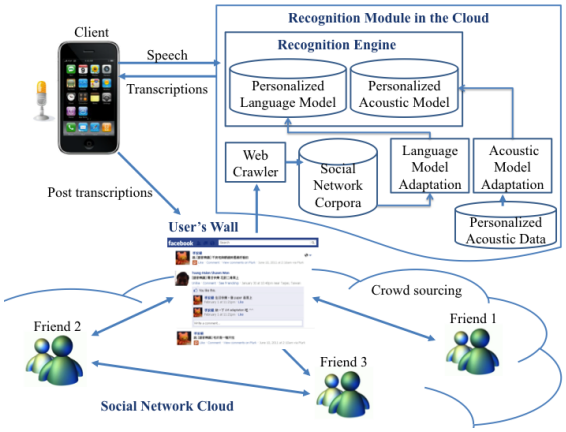

Figure 1: *The ecosystem of our crowdsourcing-based voice access of cloud application. A speaker-dependent recognizer is obtained by acoustic model and language model personalization. A web crawler was implemented for collecting social posts from users for language model personalization.*

3. Proposed Approach

As shown in Fig. 2, the RNNLM [26] contain three layers: the input layer, the hidden layer, and the output layer. The input word vector $w(t)$ represents the word at time t by an 1-of-N encoding. The context vector $s(t)$ uses distributed representation for arbitrary length histories at time t, with a recurrent connection taking the time-delayed context vector $s(t-1)$ into consideration. The output layer $y(t)$ then generates the probability distribution of the next word. In order to provide complementary information such as part-of-speech tags, topic information, or morphological information to the input vector $w(t)$, a context-dependent RNNLM variant [28] adds an additional feature layer $f(t)$ to the network in which it connects to both hidden and output layer. Therefore, all the weights needed to be learned in the network are matrices $\mathcal{W}$, $\mathcal{F}$, $\mathcal{S}$, $\mathcal{G}$, and $\mathcal{O}$. The learning process maximizes the likelihood of training data by backpropagation through time (BPTT) algorithm. Usually, a validation set is also provided to control the training epochs and learning rates.

3.1. RNNLM Personalization

Considering the fact that the collected personal corpora is small, it is impossible to train a standalone RNNLM by using only this small amount of training data. As a consequence, we fall back to the idea of language model adaptation [10]. The basic framework for language model adaptation takes two text corpora into consideration: the adaptation corpus A, which is in-domain or updated with respect to the target recognition task, but probably small and insufficient to train a robust standalone LM; the other is a large background corpus B which may not be sufficiently related to the target task or perhaps out-of-dated. Then in traditional N-gram models, the two corpora are interpolated by some methods estimating the interpolation weights between them [35, 36, 37, 38]. But for NN-based LMs, there's still not many literatures focus on the problem of adaptation. Unlike previous works [24, 39] proposed to add an additional adaptation layer between projection and input layer in feedforward NNLMs, we propose to directly fine-tune network weights by the adaptation corpus A on a RNN-based background LM. Moreover, instead of considering the user's personal corpus A and background corpus B only, we also take the corpora of his friends' C into consideration. The following three steps describe how we train and adapt the personalized RNNLMs:

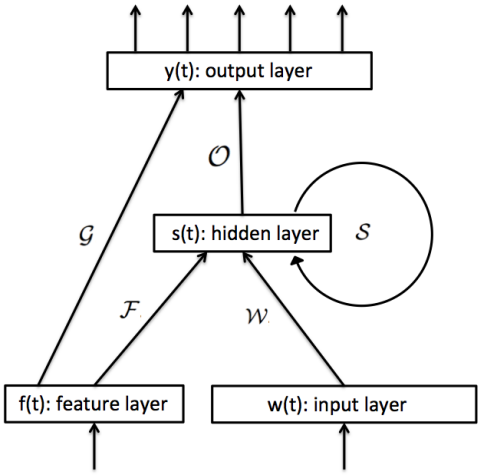

Figure 2: *The structure of context-dependent recurrent neural network language models.*

1. Train a general domain RNNLM by the background corpus B. The corpus B is split into training set and validation set such that the training set likelihood is maximized and validation set is used to control the training epochs. An initial set of model parameters $\mathcal{W}_0$, $\mathcal{F}_0$, $\mathcal{S}_0$, $\mathcal{G}_0$, and $\mathcal{O}_0$ are obtained here.
2. Given a target user's corpus A, we split it into training set T_a and validation set V_a. Copy one background RNNLM, use BPTT to fine-tune parameters $\mathcal{W}_0$, $\mathcal{F}_0$, $\mathcal{S}_0$, $\mathcal{G}_0$, and $\mathcal{O}_0$ by maximizing the likelihood of the training set T_a while again controlling the epochs by the validation set V_a. After fine-tuning, the adapted model parameters $\mathcal{W}'$, $\mathcal{F}'$, $\mathcal{S}'$, $\mathcal{G}'$, and $\mathcal{O}'$ can be attained.
3. Given friends' corpora C, we directly treat C as a compete training set. Again maximizing its likelihood but still controlling the epochs by V_a. The final model $\mathcal{W}''$, $\mathcal{F}''$, $\mathcal{S}''$, $\mathcal{G}''$, and $\mathcal{O}''$ are outputted afterwards.

There're several benefits by adopting this three-step RNNLM personalization: First, due to the common computational overhead found in RNNLMs, the background RNNLM forms a solid foundation for future model fine-tuning in which it learns some general domain histories beforehand. Secondly, although the amount of training data is small, it is believed that the distributed representation of NNLMs can amplify the training efficiency since one training sentence can inform the model combinatorial number of other sentences in the continuous space.

Thirdly, unlike our previous approach [18] computes similarities as interpolation weights when incorporating multiple friends' corpora, the use of personal validation set V_a in Step. 3 makes the approach completely data-driven. We no more worried about how to select and weight those friends but only maximize the training data we can see while at the same time use the personal validation set V_a to guarantee the adaptation orients toward the right direction. Lastly, considering the recurrent connection and the additional feature layer have already made the RNNLM structure complex, we keep the model as simple as possible without adding any additional layers.

3.2. User Oriented Features

As shown previously in Fig. 2, an additional feature layer $f(t)$ was added into the context-dependent RNNLM [28] in order to provide complementary information for the input word vector. In a previous study [29], adding complementary features such as part-of-speech tags, word lemmas, and topics, can reduce both the model perplexity and word error rate. Besides the features mentioned above, the use of language is usually influenced by the demographic characteristics of the individual, his preferences, interests, or even the role he played in a community. For example, a male user may mention more about super hero movies than a female user. People who often work together may share same jargons. This kind of user oriented features are hard to obtain in traditional text corpus, but is relatively easy from social networks.

3.2.1. Demographic information

Consider one particular demographic information that can separate users p into K clusters, we can then compute the word distribution $P(w|C_k)$ over those K clusters by

$$P(w|C_k) = \frac{\sum_{p \in C_k} n(w,p)}{\sum_k \sum_{p \in C_k} n(w,p)} \quad (1)$$

where $n(w,p)$ is the number of occurence for word w in the corpus of user p. The resulting word distribution $P(w|C_k)$ is then fed into the feature layer $f(t)$ as shown in Fig. 2. We can simply compute and cascade several demographic feature vectors in the same way to obtain richer information.

3.2.2. Latent user group

Often times, the use of language is not so directly related to the demographic characteristics of a person. Instead, his interests, habits, idols he admired, the community he joined, the friends he linked with ... etc. play an even important role [40, 41]. However, modeling all these phenomenons is difficult not only because that so many aspects make the feature dimension too big to efficiently train the RNNLMs, but also due to they are elusive so that explicit modeling is impractical. In order to efficiently model these elusive aspects implicitly, we use the method called latent factor models [4, 42]. First, we construct a user-word matrix M from those user corpora that we have collected. For simplicity, we consider only unigrams. Then, a singular value decomposition is conducted to find latent factors in the matrix. By keeping only the top-K eigenvalues, we can project both users and words into a latent space of dimension K with each dimension represents one latent factor the user or word possesses. Those latent factor feature of words are then fed into the feature layer. Note that although our method is similar to latent semantic analysis [42], there's one fundamental difference that LSA factorizes document-word matrix to find latent topics in the documents while our method factorizes user-word matrix to find any possible correlations among them.

4. Experiments

4.1. Experimental Setup

4.1.1. Corpus & LMs

Our experiments are conducted on a crawled Facebook dataset. There are a total of 42 users who logged in and authorized this project to collecting their messages and basic information for the purpose of academic research. These 42 users were treated as our target users and we tried to build a personalized language model for each of them. For each target user, 3/5 of his corpus is taken as the training set, 1/5 as the validation set, and the rest 1/5 as testing data for computing the perplexity. Furthermore, with their consents, the public data that are observable to these 42 target users are also available to our system. Through this process, besides the personal corpora for the 42 target users, we also had a whole set of publicly observable data, which are primarily data for those individuals linked with the 42 users on the network. In this way, we had 93K anonymous personal corpora, totally 3.3M sentences collected. But after preprocessing and filtering the number of sentences used in the work was approximately 2.4M. The number of sentences for each user range from 1 to 8566 with mean 25.66, with 10.59 words (Chinese or English or mixed) per sentence in average. On the network, each target user has an average of 238 friends. For the background language model, 250M sentences were collected from another popular social network site called Plurk. There were both Chinese and English words in the Plurk data with mixing rate 9:1. Modified Kneser-Ney algorithm [43] was used for N-gram language model smoothing. 46K Chinese words and the most frequent 18K English words appearing in the corpus were selected to form the lexicon. The SRILM [44] toolkit was used for N-gram LM training and adaptation. For RNNLMs, we use rnnlm toolkit [45] for implementation. We used three different feature sets in our experiment, they're pos tags (POS), demographic features (DE) considering gender (male, female, unknown) and language settings (Tradition Chinese, Simplified Chinese, English, unknown), and latent user groups (LUG) with 100 latent dimensions. We reported not only the results of using them independently but combination of them.

4.1.2. N-best rescoring

For n-best rescoring experiments, we used lattices produced by HTK toolkit [46] to generate 1000-best lists for rescoring. The LMs we used to generate n-best lists were 3-gram models also adapted by personal corpus and friend corpora with Kneser-Ney smoothing (KN3). The Mandarin tri-phone acoustic models used for first-pass decoding were trained on the ASTMIC corpus with 37 Chinese phone set, while the English tri-phone acoustic models were trained on the Sinica Taiwan English corpus with 35 English phone set, both training sets including hundreds of speakers. The acoustic models were also adapted by unsupervised MLLR speaker adaptation. We report both the results with/without speaker adaptation on two different test sets, Set *(I)* and Set *(II)*. The users in Set *(I)* were a subset of users in Set *(II)* in which the 42 target users with richer data were selected. Table. 1 summarized the comparison of the two test sets. Word error rates reported below was averaged over all sentences. The decode weights of 8.0 and 0.5 for language and acoustic models respectively were set empirically.

Table 1: Summary of two test sets for n-best rescoring.

	# of user	speakers	# of utt.	record env.
Set (I)	12	Two	840	Clean
Set (II)	42	Multiple	948	Various

Table 2: Perplexity results. Both RNNLMs and Kneser-Ney 3grams (KN3) trained with background corpus only (*Back.*), further adapted with personal corpus (*B+S*), and even further adapted with friend corpora (*B+S+F*) are shown. Several combination of hidden layer sizes (100H, 200H, 500H) and feature settings are also reported. Feature used are defined in Sec. 4.1.1.

Model		Perplexity
(a)	KN3, Back.	343.57
	KN3, B+S	299.32
	KN3, B+S+F	233.20
(b)	RNN 100H, Back.	309.50
	RNN 200H, Back.	289.14
	RNN 500H, Back.	267.38
(c)	RNN 500H, B+S	234.55
	RNN 500H, B+S+F	209.56
(d)	RNN 500H, B+S+F, POS	196.50
	RNN 500H, B+S+F, DE	199.23
	RNN 500H, B+S+F, LUG	195.59
	RNN 500H, B+S+F, POS+DE	195.76
	RNN 500H, B+S+F, POS+LUG	192.85
	RNN 500H, B+S+F, ALL	192.83
(e)	RNN 500H, B+S+F, POS + KN3	158.45
	RNN 500H, B+S+F, LUG + KN3	157.23
	RNN 500H, B+S+F, ALL + KN3	155.40

4.2. Experimental Results

4.2.1. Perplexity

Table. 2 shows the preliminary perplexity experiments on our proposed approach. Both RNNLMs and Kneser-Ney 3grams (KN3) trained with background corpus only (*Back.*), further adapted with personal corpus (*B+S*), and even further adapted with friend corpora (*B+S+F*) are shown. Several combination of hidden layer sizes (100H, 200H, 500H) and feature settings are also reported. It's not so surprising that the adaptation works well both on RNNLMs and N-grams (*B+S+F* $<$ *B+S* $<$ *Back.* in (a)(c)) considering the mismatch between the background corpus and each individual. RNNLMs with three different hidden layer sizes and different feature settings are also reported. Generally speaking, RNNLMs with larger hidden layer perform better than smaller ones (500H $<$ 200H $<$ 100H in (b)) due to its power of representing richer histories. Adding additional features can get better perplexity performance ((d) $<$ (c)). Although different features seem to provide the model different capabilities of predicting next word, but they doesn't vary too much in perplexity measure either using only one set of feature solely or combining them. This may due to the fact that models with large hidden layer can learn to compensate those information contained in different set of features by its stronger context. Lastly, as many previous studies had addressed, RNNLMs, in general, outperforms N-grams in perplexity, but will perform even better if we combine them together ((e) $<$ (a)(d)). For our best model (RNN 500H, *B+S+F*, ALL + KN3), we reduce the perplexity by 54.7% compared to background N-grams (KN3, Back.), around 41.8% compared to its simplest RNNLM version (RNN 500H, *Back.*, 500H).

4.2.2. Rescoring

Table. 3 reports the selected n-best rescoring results using our proposed methods on two different test sets, Set (I) and Set(II), with/without acoustic model adaptation respectively. We decoded our first-pass recognition results by three different 3-gram LMs, they were models trained on background corpus only (KN3 Back.), adapted by personal corpus (KN3 B+S), and further adapted by friend corpora (KN3 B+S+F). We can find that undoubtedly, the adapted models performed better (KN3 B+S+F $<$ KN3 B+S $<$ Back.). We used the best adapted N-gram models (KN3 B+S+F) to generate lattices, and 1000-best lists were extracted from the lattices, which were then used in the n-best rescoring experiments. The RNNLMs used for rescoring were all configured to 500 hidden layers while adding different types of features. The interpolation weight between RNNLM and KN3 were empirically set to 0.75. Generally, all rescoring results were better than the first-pass results (RNN KN3, RNN $<$ First-pass), which indicated that the personalized RNNLMs perform better than personalized N-grams. However, when talking about whether to combine RNNLMs with N-grams, Set (I) and Set (II) showed different opinions in which we got better results on Set (I) when combining them together but got worse when evaluated on Set (II). This may be due to the weight 0.75 is not suitable for Set (II), which indicated that a better weight selection mechanism should be carefully surveyed. We also found out that more features doesn't necessarily give better results (others $<$ RNN POS-LUG). Usually, single use of pos tags (POS) and latent user group (LUG) features gave more improvements compare to other settings.

Table 3: The selected n-best rescoring results conducted on two different test sets, Set (I) and Set (II), with/without adapted acoustic model (MLLR/AM) respectively.

WER(%)		Set (I)		Set (II)	
		AM	MLLR	AM	MLLR
First-pass	KN3 Back.	40.98	38.75	49.19	43.80
	KN3 B+S	40.31	38.08	48.63	43.39
	KN3 B+S+F	33.45	30.13	46.81	41.95
RNN 500H	RNN None	32.59	29.80	44.87	40.10
	RNN POS	32.51	29.50	44.84	**40.06**
	RNN DE	32.65	29.95	44.94	40.18
	RNN LUG	32.95	29.36	**44.78**	40.17
	RNN POS-LUG	32.46	29.58	44.86	40.17
RNN 500H+KN3	RNN None	32.49	29.37	45.31	40.57
	RNN POS	32.53	**29.26**	45.23	40.38
	RNN DE	**32.37**	29.91	45.47	40.40
	RNN LUG	32.51	**29.26**	45.35	40.53
	RNN POS-LUG	32.75	29.75	45.32	40.52

5. Conclusion

In this paper, we propose a new paradigm for RNNLM personalization in which social network data were crawled from a crowdsourcing platform as adaptation resources. A three-step adaptation mechanism considering feasibility and complexity was also proposed. Some user oriented features were also incorporated into the model for better word prediction capabilities. Experiments showed that by personalized RNNLMs while adding user oriented features, the perplexity can be reduced up to 54.7% compared to background N-grams and 41.8% compared to original RNNLM. Moreover, the 1000-best rescoring experiments also reported 2.9% to 4.5% relative word error rate reductions on different experimental settings.

6. References

[1] G. Zweig and C. Shuang yu, "Personalizing model m for voicesearch," in *Proc. on InterSpeech*, 2011.

[2] M. Speretta and S. Gauch, "Personalized search based on user search histories," in *Proc. on Web Intelligence*, 2005.

[3] Y. H. Cho, J. K. Kim, and S. H. Kim, "A personalized recommender system based on web usage mining and decision tree induction," *Expert Systems with Applications*, 2002.

[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, 2009.

[5] F. Walter, S. Battiston, and F. Schweitzer, "A model of a trust-based recommendation system on a social network," *Autonomous Agents and Multi-Agent Systems*, 2008.

[6] M.-H. Park, J.-H. Hong, and S.-B. Cho, "Location-based recommendation system using bayesian users preference model in mobile devices," in *Ubiquitous Intelligence and Computing*, 2007.

[7] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech and Language*, 1995.

[8] P. C. Woodland, "Speaker adaptation for continuous density hmms: A review," in *Proc. on ITRW on Adaptation Methods for Speech Recognition*, 2001.

[9] l. G. Jean and C.-H. Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *IEEE Transactions on Speech and Audio Processing*, 1994.

[10] J. R. Bellegarda, "Statistical language model adaptation: review and perspectives," *Speech Communication*, 2004.

[11] A. Heidel and L.-S. Lee, "Robust topic inference for latent semantic language model adaptation," in *Proc. on ASRU*, 2007.

[12] H. Bo-June and J. Glass, "Style and topic language model adaptation using hmm-lda," in *Proc. on EMNLP*, 2006.

[13] D. Hakkani-Tur, G. Tur, and L. Heck, "Research challenges and opportunities in mobile applications," *Signal Processing Magazine, IEEE*, 2011.

[14] X. Sun and A. May, "The role of spatial contextual factors in mobile personalization at large sports events," *Personal and Ubiquitous Computing*, 2009.

[15] S. Arbanowski, P. Ballon, K. David, O. Droegehorn, H. Eertink, W. Kellerer, H. van Kranenburg, K. Raatikainen, and R. Popescu-Zeletin, "I-centric communications: personalization, ambient awareness, and adaptability for future mobile services," *Communications Magazine*, 2004.

[16] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational Linguistics*, 1992.

[17] A. Stolcke, "Entropy-based pruning of backoff language models," in *Proc. of the DARPA Broadcast News Transcription and Understanding Workshop*, 2000.

[18] T.-H. Wen, H.-Y. Lee, T.-Y. Chen, and L.-S. Lee, "Personalized language modeling by crowd sourcing with social network data for voice access of cloud applications," in *Proc. on IEEE SLT workshop*, 2012.

[19] J. Wu and S. Khudanpur, "Combining nonlocal, syntactic and n-gram dependencies in language modeling," in *Proc. on EuroSpeech*, 1999.

[20] S. Khudanpur and J. Wu, "Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling," *Computer Speech and Language*, 2000.

[21] H. S. Le, A. Allauzen, and Y. Fran, "Measuring the influence of long range dependencies with neural network language models," in *Proc. on NAACL-HLT Workshop*, 2012.

[22] I. Oparin, M. Sundermeyer, H. Ney, and J. Gauvain, "Performance analysis of neural networks in combination with n-gram language models," in *Proc. on ICASSP*, 2012.

[23] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, 2003.

[24] J. Park, X. Liu, M. J. F. Gales, and P. C. Woodland, "Improved neural network based language modeling and adaptation," in *Proc. on InterSpeech*, 2010.

[25] H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Structured Output Layer neural network language model," in *Proc. on ICASSP*, 2011.

[26] T. Mikolov, M. Karafit, L. Burget, J. Cernock, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. on InterSpeech*, 2010.

[27] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proc. on ICASSP*, 2011.

[28] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *Proc. on IEEE SLT workshop*, 2012.

[29] Y. Shi, P. Wiggers, and C. M. Jonker, "Towards recurrent neural networks language models with linguistic and contextual features," in *Proc. on InterSpeech*, 2012.

[30] A. Doan, R. Ramakrishnan, and A. Y. Halevy, "Crowdsourcing systems on the world-wide web," *Communications of the ACM*, 2011.

[31] Munro and Robert, "Crowdsourcing and language studies: the new generation of linguistic data," in *Proc. on NAACL*, 2010.

[32] K. Berberich, S. Bedathur, O. Alonso, and G. Weikum, "A language modeling approach for temporal information needs," in *Advances in Information Retrieval*, 2010.

[33] J. Liu, S. Cyphers, P. Pasupat, I. McGraw, and J. Glass, "A conversational movie search system based on conditional random field," in *Proc. on InterSpeech*, 2012.

[34] I. McGraw, S. Cyphers, P. Pasupat, J. Liu, and J. Glass, "Automating crowd-supervised learning for spoken language systems," in *Proc. on InterSpeech*, 2012.

[35] R. Iyer and M. Ostendorf, "Modeling long distance dependence in language: topic mixtures versus dynamic cache models," *IEEE Transactions on Speech and Audio Processing*, 1999.

[36] A. Heidel, H.-A. Chang, and L.-S. Lee, "Language model adaptation using latent dirichlet allocation and an efficient topic inference algorithm," in *Proc. on InterSpeech*, 2007.

[37] M. Federico, "Efficient language model adaptation through mdi estimation," in *Proc. on EuroSpeech*, 1999.

[38] C. Chelba and F. Jelinek, "Structured language modeling," *Computer Speech and Language*, 2000.

[39] X. Liu, M. J. F. Gales, and P. C. Woodland, "Improving lvcsr system combination using neural network language model cross adaptation," in *Proc. on InterSpeech*, 2011.

[40] J. Paolillo, "The virtual speech community: Social network and language variation on irc," *Journal of Computer-Mediated Communication*, 1999.

[41] D. Rosen and M. Corbit, "Social network analysis in virtual environments," in *Proc. on ACM Hypertext*, 2009.

[42] T. K. Landauer, P. W. Foltz, and D. Laham, "An Introduction to Latent Semantic Analysis," *Discourse Processes*, 1998.

[43] F. James, "Modified kneser-ney smoothing of n-gram models modified kneser-ney smoothing of n-gram models," Tech. Rep., 2000.

[44] A. Stolcke, "Srilm - an extensible language modeling toolkit," in *Proc. on Spoken Language Processing*, 2002.

[45] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocky, "Rnnlm - recurrent neural network language modeling toolkit," in *Proc. on ASRU*, 2011.

[46] S. J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, *The HTK Book Version 3.4*. Cambridge University Press, 2006.