COMPARISON OF DATA AUGMENTATION AND ADAPTATION STRATEGIES FOR CODE-SWITCHED AUTOMATIC SPEECH RECOGNITION

Min Ma, Bhuvana Ramabhadran, Jesse Emond, Andrew Rosenberg, Fadi Biadsy

Google Inc., USA
{minm, bhuv, emond, rosenberg, biadsy}@google.com

ABSTRACT

Code-switching occurs when the speaker alternates between two or more languages or dialects. It is a pervasive phenomenon in most Indic spoken languages. Code-switching poses a challenge in language modeling as it complicates the orthographic realization of text, and generally, there is a shortage of code-switched data. In this paper, we investigate data augmentation and adaptation strategies for language modeling. Using Bengali and English as an example, we study augmenting the code-switched transcripts with separate transliterated Bengali and English corpora. We present results on two speech recognition tasks, namely, voice search and dictation. We show improvements on both tasks with Maximum Entropy (MaxEnt) and Long Short-Term Memory (LSTM) language models (LMs). We also explore different adaptation strategies for MaxEnt LM and LSTM LM, demonstrating that the transliteration-based data-augmented LSTM LM matches the adapted MaxEnt LM which is trained on more Bengali-English data.

Index Terms— data augmentation, language model adaptation, code-switched automatic speech recognition

1. INTRODUCTION

Code-switching[1] is prevalent in many multilingual communities, wherein a speaker alternates between two or more languages, or language varieties. It is very common in most Indic languages, as many Indic speakers are at least trilingual, speaking various combinations of their native language with words borrowed from the more commonly-spoken Hindi and English languages. For example, it is common to see combinations such as Bengali-English, Bengali-Hindi, and Bengali-Hindi-English in daily speech. Naturally, this leads to the same word being transcribed differently under different writing systems. Given that all forms of these transcriptions are correct, consistent normalization of code-switched text becomes crucial for statistical models.

Two threads of research have appeared in the literature for addressing these challenges. The first approach uses multi-pass speech recognition. Here, regions of code-switching are first identified using acoustics based language identification methods and then rescored with corresponding monolingual acoustic and language models [2, 3, 4]. The second approach is the use of multilingual acoustic and language models in a single recognition pass [5, 6]. Both approaches have limitations. While the former approach requires several passes and depends on the quality of the language identification system, the latter approach requires linguistic expertise.

[1]Some linguists argue, code-mixing specifically refers to intra-sentential code-switching [1]. In this paper, we use code-mixing and code-switching interchangeably.

The lack of sufficient code-switched training material poses a challenge for language modeling. To address this and provide consistent normalization, a more recent approach proposed the use of transliteration [7]. It yielded significant gains in automated speech recognition (ASR) performance. In this paper, we present a generic approach for training language models for ASR that can leverage code-switched text. We use the Bengali-English language pair as an example to evaluate and analyze our approach. The approach builds on the work presented in [7] and compares data augmentation and adaptation strategies to address code-switched ASR.

2. PREVIOUS WORK

Data augmentation strategies to address the challenges of code-switching have been explored over the last couple of decades. One of the first pieces of work used machine translated text [3] to augment available code-switched text. The authors show that addition of artificially synthesized code-switched text to build the recurrent neural network (RNN) language models achieved up to 16.9% relative reduction in perplexity and a 2.7% relative improvement in mixed error rate on the SEAME corpus [8]. Inspired by the Equivalence Constraint Theory [9], Pratapa et al. [10] recently proposed to generate grammatically valid artificial code-mixing data using parallel monolingual sentences. The synthesized data helped reduce the perplexity of an RNN LM. The authors also claim that *randomly* generated code-switched data does not help to decrease perplexity of code-switched text. In [11], the authors model this linguistic equivalence constraint as a syntactic inversion constraint into a statistical code-switch language model where the language model is composed of a code-switched boundary prediction model, translation model, and a reconstruction model. This approach yielded modest gains on Chinese-English ASR.

More recently, [12] proposed to alter the structure of a cell in the RNN to include language-specific components that model code-switched text. Pretraining the LM on synthetic text from a generative model estimated using the training data, then fine-tuning it on the same training data, the authors observed up to 13% relative perplexity improvements. Other work on data augmentation has explored the use of monolingual interpolated LMs trained separately on monolingual texts [6]. In [13], the authors proposed three ways to augment code-switched Frisian-Dutch data for LMs: text generated from RNN LMs trained on speech transcripts, use of ASR transcripts, and translated data from an external corpus. They report significant reductions in perplexity and ASR performance improvements on this low-resource-high-resource language pair.

The process of converting sequences from one writing system to another, i.e., transliteration, has been used extensively in machine translation [14, 15, 16] and retrieval [17]. Transliteration of Indic

languages to Latin script and vice-versa is particularly challenging due to the presence of a large combination of consonants, vowels, and diacritics which result in non-unique mappings and non-standard spellings. The first use of transliteration to normalize code-switched text consistently for ASR was proposed in [7].

In this paper, we propose a set of data augmentation strategies based on transliteration. Different from previous work which introduces external monolingual textual resources, our experiments demonstrate the effectiveness of leveraging the code-mixed data itself using transliteration. This could be extremely meaningful for modeling low- and medium-resource languages. We also utilize an on-the-fly unsupervised method to recognize the language code of each word, avoiding the extra computational cost of building a high-quality language identification model. We compare these augmentation strategies to standard LM adaptation methods, using Maximum Entropy based LMs and LSTM LMs.

3. LANGUAGE MODELS

We take a rescoring approach to language modeling. The first pass LM is consistent across all experiments. All the proposed approaches modify the second-pass LM.

3.1. First-pass LM

The first-pass LM used to generate lattices is an ensemble of 5-gram language models trained on spoken and written texts from multiple resources. The interpolation weights for the various LMs is determined through Bayesian Interpolation [18]. A second-pass LM, either a maximum entropy based LM *or* an LSTM LM, is used to rescore the N-best lists generated from the first pass. The second-pass LM is interpolated log-linearly with the first-pass LM. Our model vocabulary contains 122K words – 89.9% are Bengali, 8.3% are English, and the rest are numbers and urls.

3.2. MaxEnt LM

In [19], a hierarchical Backoff MaxEnt LM [20] for 2nd-pass rescoring has shown significant reduction in WER for the voice search task across multiple languages. In this work, we make use of the same 2nd-pass rescoring MaxEnt LM framework described in [19]. We also employ similar features: word-level N-grams up to 5-grams, word cluster N-grams from 3- to 5-grams, skip word bigrams up to a gap of 5 words, left and right skip trigrams up to a gap of 3 words, plus backoff features. We select the most frequent one billion features for our MaxEnt model. Our model vocabulary here is the same as that of the first-pass LM. Words are clustered to 700 clusters, using an algorithm similar to [21, 22].

3.3. LSTM LM

Our LSTM LM is a 1024-node, 2-layer model that uses a 1024-dimensional word embedding features, and a 8192-node sampled softmax layer. It couples the internal input and forget gates as proposed in [23]. We add markers denoting sentence-start (<S>) and sentence-end (</S>) to each word sequence in the training corpus. The model is trained using truncated backpropagation through time (BPTT) with an unrolling of 20 time steps using a cross entropy loss between predicted words and reference word labels. Mini-batch stochastic gradient descent (SGD) [24] is used with an Adagrad optimizer [25] and a batch size of 128 sequences. We choose a learning rate of 0.2. We found it crucial to use gradient clipping on the LSTM gradients (clipping L2-norm ≤ 1.0).

4. DATA

It is widely recognized that collecting large amounts of code-switched textual data is challenging, as code-switching is rare in formal documents. Therefore, the vast majority of our training corpus is composed of code-switched speech transcripts generated by humans. We note there is a lot of variance in the writing scripts used by humans when transcribing code-switched speech. We refer to this Bengali-English code-mixed textual corpus as "CM train".

We pool an additional Bengali-English corpus (denoted as "wCM"), derived from web crawls, anonymized, written search queries, news articles and books. In addition to significant domain differences between "wCM" and "CM train", the percentages of utterances which contain English tokens are very different, as shown in Table 1. We also make use of an external Indian English corpus

Table 1. Statistics of the data sets ("Latin" refers to the percentage of utterances that contain Latin tokens).

Data Set	%(Latin)	#(utterances)
CM train	5.8	1.6 million
wCM	57.8	510 million
inEN	84.3	6.6 million
CM dev	5.2	14.7 k

of speech transcripts (denoted as "inEN"), where we find about 15% transcripts are in Bengali.

Our development set ("CM dev") is based on unsupervised data derived from real voice traffic. All the language models are optimized by minimizing the perplexity on this set.

5. LANGUAGE MODEL ADAPTATION

Efficiently making use of in-domain data to adapt language models can shift them to a space closer to the actual distribution of test data. In this section, we investigate the adaptation approaches for MaxEnt LM and LSTM LM.

5.1. MaxEnt LM Adaptation: Pre-train & Fine-tune

Biadsy et al. [19] have shown that a MaxEnt LM trained on out-of-domain data can be successfully adapted to certain domains. By leveraging a small in-domain data, the model showed substantial reduction in WER when tested on the same domain. We adopt the same pre-training and adaptation methodology in this work. We first pre-train the MaxEnt LM on the corpus that unifies written domain text and spoken domain text, as described in Section 4. Then, upon model convergence (measured on our dev set), we fine-tune this model on the in-domain data (CM train), as it closely reflects voice-search distribution. Similar to [19], we adapt the model for three iterations with the learning rates .25, .2, and .12.

5.2. LSTM LM Feature-based Adaptation

It has been found that feature-based adaptation of recurrent neural network LMs by incorporating domain-specific auxiliary features can reduce both perplexity and word error rate [26, 27]. Based on our prior work on feature-based adaptation [28], we adapt the LSTM

language models in the following manner: we build a separate embedding layer to encode the writing system information associated with each word, then concatenate this language code embedding with word embedding and feed the concatenated embedding to the hidden layers. We utilize a Unicode based language identification method. We scan each input token character by character, and decide based on the Unicode range if each character belongs to the English alphabet or Bengali one. If all the characters are English characters, we identify the token as English; otherwise, as non-English (almost all are Bengali tokens). We embed the language code signals using a 2-dimensional dense vectors.

6. EXPERIMENTAL RESULTS AND ANALYSIS

We evaluate the data augmentation and LM adaptation ideas by building baseline LMs on the corpora described in Sec. 4. The ASR system employs an LSTM acoustic model which has 5 layers, with each layer consisting of 768 LSTM cells. The acoustic models were trained with approximately 4.5k hours of Bengali-English audio using asynchronous stochastic gradient descent minimizing Connectionist Temporal Classification (CTC) [29] and state-level Minimum Bayesian Risk (sMBR) objective functions [30]. ASR performance is measured using the transliteration-optimized word error rate (*toWER*) proposed in [7] which reduces ambiguity and transcription errors. It is computed after transliterating both the reference and hypothesis to one writing system corresponding to the native locale, in this case, Bengali script.

6.1. Data Augmentation Experimental Results

To augment the language models with more training data, we propose to synthesize text by transliterating this code-mixed corpus to a monolingual space via a weighted finite state transducer (WFST) as described in [7].

We first generate two copies of the "CM train" corpus. One copy is generated by transliterating the text into Bengali writing system ("CM2BN") and the second is generated by transliterating all text to English writing system ("CM2EN"). The rationale behind this augmentation is that two monolingual transliterated copies would enrich the surface realization of our training data.

In a second form of data augmentation, we enhance our training data with the "inEN" Indian English corpus by transliterating it into the Bengali writing system ("inEN2BN"). The rationale behind this approach is based on the significant amount of Latin script seen in the training corpus. We hypothesize that this augmentation will add reliability in modeling the English word combinations.

MaxEnt LMs: We interpolate the first-pass and MaxEnt LMs by (0.81, 0.19) for the voice search (VS) and by (0.75, 0.25) for the dictation task (D). The interpolation weights are chosen empirically. Training on "CM train" yields a the baseline *toWER* of 18.4 for VS, and *toWER* of 15.4 for dictation.

As shown in Table 2, we find that by transliterating the original code-mixed transcript to Bengali space, and adding it to "CM train" ($i.e.$"CM train+CM2BN"), we reduce *toWER* from 18.4 to 18.1 (1.6% relative reduction) for voice search, while maintaining the performance on dictation.

By transliterating the external English spoken transcript corpus to Bengali space, and adding it to the original data ($i.e.$"CM train+inEN2BN"), we reduce *toWER* from 18.4 to 18.2 (1.1% relative reduction) for voice search task, indicating the reliability of actual English words supplemented by the transliterated Indian English data. The model holds the *toWER* performance on dictation.

By adding the above two transliterated data sets to the original Bengali-English mixed data, we have achieved the best *toWER* performances: it successfully reduces *toWER* by 2.2% relative for voice search, and reduces *toWER* for dictation by 1.3% relative. The improvements suggest complementary strengths from transliterated "CM train" and transliterated external English corpus. This might be particularly important, as it shows the promise of transliterating textual training data from other languages.

We find transliterating Bengali-English data to English (denoted as "CM2EN"), indicated by (5), leads to the degradation in performance relative to the baseline on both voice search and dictation. This discourages transliterating code-mixed text to the secondary language when building an LM for the imary language.

We find it harder to reduce *toWER* on dictation, we speculate that the utterance length of duration is generally longer than the length of voice search queries, and MaxEnt LM might not be able to represent long-span word dependencies effectively.

Table 2. *toWER (%) of data augmentation experiments with MaxEnt LMs. (Here "CM" is short for "CM train").*

Training Data, 2nd-pass LM	VS	D
(1) CM (baseline), MaxEnt	18.4	15.4
(2) CM+CM2BN, MaxEnt	18.1	15.4
(3) CM+inEN2BN, MaxEnt	18.2	15.4
(4) CM+CM2BN+inEN2BN, MaxEnt	**18.0**	**15.2**
(5) CM+CM2EN, MaxEnt	18.6	15.5
(6) CM+CM2BN+CM2EN, MaxEnt	18.1	15.4

LSTM LMs: Based on the results of MaxEnt LM rescoring experiments, we use the data augmentation configurations that yield the best performances in *toWER* to train LSTM LMs. These results are presented in Table 3. The LSTM LM score is interpolated with the 5-gram LM score by a weight of 0.5. We find all the LSTM LMs outperform the MaxEnt models that are trained on the same training data configuration: *toWER* has been reduced up to 4.4% relative for voice search, and reduced up to 5.2% relative for dictation. We are hesitant to conclude that LSTM LMs are inherently superior to MaxEnt LMs in any data augmentation scenarios. In fact, in this section, we describe a MaxEnt model trained with additional text from a written domain that achieves performance very close to unadapted LSTM LMs.

In many experiments in this paper, and reported elsewhere in the literature, the differences between MaxEnt and LSTM performance is relatively minor. This suggests a number of possible explanations. First, the data-augmented MaxEnt may require additional hyperparameter tuning to achieve the same gains observed in LSTM models by the baseline training approach. For example, when training on augmented data, the MaxEnt model may require additional parameters. When we add transliterated copies of CM train, we double the count of n-grams containing only Bengali words. This may bias count-based models (N-gram and MaxEnt) more than LSTMs. That said, it is also possible that the LSTM is able to model something unique about bilingual data by being exposed to essentially parallel texts. This may allow the LSTM to learn a code-independent internal representation aggregating distinct writing forms of the same underlying tokens.

6.2. LM Adaptation Experimental Results

We conduct a range of adaptation experiments, as an alternative to exploit the full potential of in-domain data, and compare them to

Table 3. *toWER (%) of data augmentation experiments with LSTM LMs. "(1)", "(2)", "(4)" refer to equivalent data configurations in Table 2.*

Training Data, 2nd-pass LM	VS	D
(1) CM, LSTM	18.3	15.1
(2) CM+CM2BN, LSTM	**17.3**	14.7
(4) CM+CM2BN+inEN2BN, LSTM	17.6	**14.6**

the non-adapted LMs which are trained on transliteration-augmented data.

MaxEnt LM Adaptation: We find large *toWER* reduction from adapting on "CM train". Training on the Bengali-transliterated corpus which pools code-switched spoken transcripts and code-switch written domain text ($i.e.$ first row in Table 4) only gets 19.8% *toWER* in VS, and 15.4% in dictation. Training on the same data, but adding a fine-tuning phase to adapt MaxEnt LM on the spoken transcript dramatically reduces *toWER* to 19.8% in VS (12.6% rel. reduction), and 14.4% (6.5% rel. reduction) in dictation. This suggests a significant difference between the written and spoken data, and, moreover, the mismatch is amplified when rescoring hypotheses of VS, but not that obvious for rescoring hypotheses of dictation.

LSTM LM Adaptation: We explore the potential of LSTM LMs on a smaller set ($i.e.$ "CM train" only). We find absolute reductions of 0.1% and 0.2% in *toWER* by adapting LSTM LMs, compared to their non-adapted LSTM peers, on both VS and dictation. As explicit language code information is present to LSTM LMs, richer contextual signals may be captured, leading to improvements over their unadapted counterparts.

As shown in Table 4, we find close performance in *toWER* of adapted MaxEnt LM (17.3% and 14.4%) and adapted LSTM LM (17.2% and 14.6%), for both voice search and dictation. We note this MaxEnt LM was pre-trained on a larger training set ("CM train + wCM") before adaptation, while the LSTM LM was only adapted on its spoken transcript portion and its Bengali-transliterated copy ("CM2BN"), by learning the language code information. The difference in the amount of training and adaptation data might result in the regression on the dictation task (14.4% vs. 14.6%). By introducing a new data source, $i.e.$ English speech transcripts which have been transliterated into Bengali, we compensate for the loss in *toWER* of dictation from 14.6% to 14.5%.

Table 4. *Adaptation results with MaxEnt and LSTM LMs (prefix "ada" refers to "adapted"), in toWER (%).*

Data, 2nd-pass LM	VS	D
(7) (CM+wCM)2BN, MaxEnt	19.8	15.4
(7) (CM+wCM)2BN, adaMaxEnt	**17.3**	**14.4**
(1) CM, adaLSTM	18.2	15.3
(2) CM+CM2BN, adaLSTM	**17.2**	14.6
(4) CM+CM2BN+inEN2BN, adaLSTM	17.4	**14.5**

6.3. Analysis of Error types

Inspecting into the detailed error types (deletion, insertion, substitution), we find that the unadapted MaxEnt LM does well in controlling insertion errors, while MaxEnt adaptation provides reductions to deletions and substitutions, while increasing the number of insertions (as shown in Table 5). This is consistent with a hyperparameter setting that has somewhat reduced "coverage", but increased model fidelity in the covered regions. This error pattern is not observed in LSTM models: we see some minor improvements to deletion errors through adaptation with virtually no change to insertions and substitutions. This may suggest that the adaptation via a language code enables it to effectively increase its coverage, but this does very little to improve insertions and substitutions. Comparing across modeling technique, the unadapted LSTM LM introduces slightly more deletion errors than the unadapted MaxEnt LM, however, these differences are compensated via LSTM adaptation.

Table 5. *toWER (%) of each error type for different LMs. "(4)" stands for "CM+CM2BN+inEN2BN" data configuration, while "(7)" stands for "(CM+wCM)2BN" data configuration.*

Data, 2nd-pass LM	VS (del/ins/sub)	D (del/ins/sub)
(7), MaxEnt	6.7/**1.3**/11.8	3.2/**1.5**/10.6
(7), adaMaxEnt	4.7/1.9/10.7	2.6/1.8/10.0
(4), MaxEnt	3.8/2.5/11.8	1.8/2.6/10.8
(4), LSTM	4.0/2.3/11.3	1.9/2.5/10.2
(4), adaLSTM	**3.7/2.3/11.3**	**1.8/2.5/10.1**

7. CONCLUSIONS

Code-switching poses a particular set of challenges for language modeling. There is a combinatorial explosion of valid contexts to condition a token, and the token itself can be realized in multiple languages.

In this paper, we present a simple yet effective transliteration-based data augmentation approach to improving speech recognition performance of code-switched Bengali-English. By conducting a range of language model rescoring experiments with MaxEnt and LSTM models, we demonstrate its effectiveness in reducing the recognition error rate of Bengali-English speech. Transliterating the code-mixed textual corpus to the primary language (Bengali in this case) and adding it to training data significantly reduces the *toWER*, especially for LSTM LMs. We also compare this approach to adaptation. We find that this simple augmentation performs equally well as an adapted MaxEnt LM which was pre-trained on a much larger set of code-switched data. LSTM adaptation results in modest but consistent gains. We find that this augmentation approach is more easily applied to LSTM than MaxEnt modeling. The interaction between data augmentation and MaxEnt hyperparameters remains a question for future investigation.

This work has shown clear gains in modeling Bengali-English code-switched data. While there is nothing specific that limits the approach to these languages, the robustness of this approach to other language pairs (or groups of more than two languages) has not been assessed here.

8. REFERENCES

[1] Rabia Redouane, "Linguistic constraints on codeswitching and codemixing of bilingual moroccan arabic-french speakers in canada," in *ISB4: Proceedings of the 4th International Symposium on Bilingualism*, 2005, pp. 1921–1933.

[2] Dau-Cheng Lyu and Ren-Yuan Lyu, "Language identification on code-switching utterances using multiple cues," in *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[3] Ngoc Thang Vu, Dau-Cheng Lyu, Jochen Weiner, Dominic Telaar, Tim Schlippe, Fabian Blaicher, Eng-Siong Chng, Tanja Schultz, and Haizhou Li, "A first speech recognition system for Mandarin-English code-switch conversational speech," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2012, pp. 4889–4892.

[4] Basem HA Ahmed and Tien-Ping Tan, "Automatic speech recognition of code switching speech using 1-best rescoring," in *International Conference on Asian Language Processing (IALP)*. IEEE, 2012, pp. 137–140.

[5] Dau-Cheng Lyu, Ren-Yuan Lyu, Yuang-chin Chiang, and Chun-Nan Hsu, "Speech recognition on code-switching among the Chinese dialects," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2006, vol. 1, pp. I–I.

[6] Kiran Bhuvanagirir and Sunil Kumar Kopparapu, "Mixed language speech recognition without explicit identification of language," *American Journal of Signal Processing*, vol. 2, no. 5, pp. 92–97, 2012.

[7] Jesse Emond, Bhuvana Ramabhadran, Brian Roark, Pedro Moreno, and Min Ma, "Transliteration based approaches to improve code-switched speech recognition performance," in *Proc. SLT*. IEEE, 2018.

[8] Heike Adel, Ngoc Thang Vu, Franziska Kraus, Tim Schlippe, Haizhou Li, and Tanja Schultz, "Recurrent neural network language modeling for code switching conversational speech," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2013, pp. 8411–8415.

[9] Shana Poplack, "Sometimes ill start a sentence in spanish y termino en espanol: toward a typology of code-switching1," *Linguistics*, vol. 18, no. 7-8, pp. 581–618, 1980.

[10] Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali, "Language modeling for code-mixing: The role of linguistic theory based synthetic data," in *Association for Computational Linguistics*. 2018, pp. 1543–1553, Association for Computational Linguistics.

[11] Ying Li and Pascale Fung, "Code-switch language model with inversion constraints for mixed language speech recognition," *Proceedings of COLING 2012*, pp. 1671–1680, 2012.

[12] Saurabh Garg, Tanmay Parekh, and Preethi Jyothi, "Code-switched language models using dual rnns and same-source pretraining," *arXiv preprint arXiv:1809.01962*, 2018.

[13] Emre Yılmaz, Henk van den Heuvel, and David A van Leeuwen, "Acoustic and textual data augmentation for improved asr of code-switching speech," *arXiv preprint arXiv:1807.10945*, 2018.

[14] Kevin Knight and Jonathan Graehl, "Machine transliteration," *Computational linguistics*, vol. 24, no. 4, pp. 599–612, 1998.

[15] Nadir Durrani, Hassan Sajjad, Alexander Fraser, and Helmut Schmid, "Hindi-to-Urdu machine translation through transliteration," in *Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 465–474.

[16] Ahmad Musleh, Nadir Durrani, Irina Temnikova, Preslav Nakov, Stephan Vogel, and Osama Alsaad, "Enabling medical translation for low-resource languages," *arXiv preprint arXiv:1610.02633*, 2016.

[17] Paola Virga and Sanjeev Khudanpur, "Transliteration of proper names in cross-lingual information retrieval," in *Proceedings of the ACL 2003 workshop on Multilingual and mixed-language named entity recognition-Volume 15*. Association for Computational Linguistics, 2003, pp. 57–64.

[18] Mehryar Mohri, Fernando Pereira, and Michael Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[19] Fadi Biadsy, Mohammadreza Ghodsi, and Diamantino Caseiro, "Effectively building tera scale MaxEnt language models incorporating non-linguistic signals," *Proc. Interspeech 2017*, pp. 2710–2714, 2017.

[20] Fadi Biadsy, Keith Hall, Pedro J Moreno, and Brian Roark, "Backoff inspired features for maximum entropy language models," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[21] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[22] Jakob Uszkoreit and Thorsten Brants, "Distributed word clustering for large scale class-based language modeling in machine translation," *Proceedings of ACL-08: HLT*, pp. 755–762, 2008.

[23] Haşim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[24] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola, "Efficient mini-batch training for stochastic optimization," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 661–670.

[25] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[26] Tomas Mikolov and Geoffrey Zweig, "Context dependent recurrent neural network language model.," *SLT*, vol. 12, no. 234-239, pp. 8, 2012.

[27] Salil Deena, Raymond WM Ng, P Madhyashta, Lucia Specia, and Thomas Hain, "Semi-supervised adaptation of rnnlms by fine-tuning with domain-specific auxiliary features," in *Proceedings of INTERSPEECH 2017: Conference of the International Speech Communication Association*. ISCA, 2017, pp. 2715–2719.

[28] Min Ma, Shankar Kumar, Fadi Biadsy, Michael Nirschl, Tomas Vykruta, and Pedro Moreno, "Modeling non-linguistic contextual signals in lstm language models via domain adaptation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6094–6098.

[29] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*. ACM, 2006.

[30] Brian Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2009, pp. 3761–3764.