An Investigation of Code-Switching Attitude Dependent Language Modeling

Ngoc Thang Vu, Heike Adel and Tanja Schultz

Institute for Anthropomatics, Karlsruhe Institute of Technology (KIT)
thang.vu@kit.edu, heike.adel@student.kit.edu, tanja.schultz@kit.edu

Abstract. In this paper, we investigate the adaptation of language modeling for conversational Mandarin-English Code-Switching (CS) speech and its effect on speech recognition performance. First, we investigate the prediction of code switches based on textual features with focus on Part-of-Speech (POS) tags. We show that the switching attitude is speaker dependent and utilize this finding to cluster the training speakers into classes with similar switching attitude. Second, we apply recurrent neural network language models which integrate the POS information into the input layer and factorize the output layer into languages for modeling CS. Furthermore, we adapt the background N-Gram and RNN language model to the different Code-Switching attitudes of the speaker clusters which lead to significant reductions in terms of perplexity. Finally, using these adapted language models we rerun the speech recognition system for each speaker and achieve improvements in terms of mixed error rate.

Keywords: multilingual speech processing, code switch attitude, language model adaptation

1 Introduction

Code-Switching (CS) speech is defined as speech that contains more than one language ('code'). It is a common phenomenon in multilingual communities where people of different cultures and language background communicate with each other [2]. The switch between languages can happen between or within an utterance. In this paper, we show that the decision whether and when a speaker changes the language is rather individual ('Code-Switching attitude').
For the automated processing of spoken communication in these scenarios, a speech recognition system must be able to handle code switches. However, the components of speech recognition systems are usually trained on monolingual data, particulary when there is a lack of multilingual training data. This is why the CS task appears to be difficult to solve.
While there have been promising research results in the area of acoustic modeling to handle Code-Switching, only few approaches so far address this challenge in the language model. Recently, it has been shown that recurrent neural network language models (RNNLMs) improve perplexity and error rates in speech recognition systems in comparison to traditional N-Gram approaches [9, 10, 15].

One reason for that is their ability to handle longer contexts. Furthermore, the integration of additional features as input is rather straight-forward due to their structure. Recently, we proposed an extended structure of recurrent neural networks in order to predict CS points. In this paper, we extend this work by showing that Code-Switching can be regarded as a speaker dependent phenomenon. Hence, it is possible to cluster speakers with similar Code-Switching attitudes to obtain more specific models. Our experimental results demonstrate that this clustering leads to significant improvements in terms of perplexity for each test speaker and that these improvements also transform into error rate reductions. Figure 1 illustrates our adaptation process.

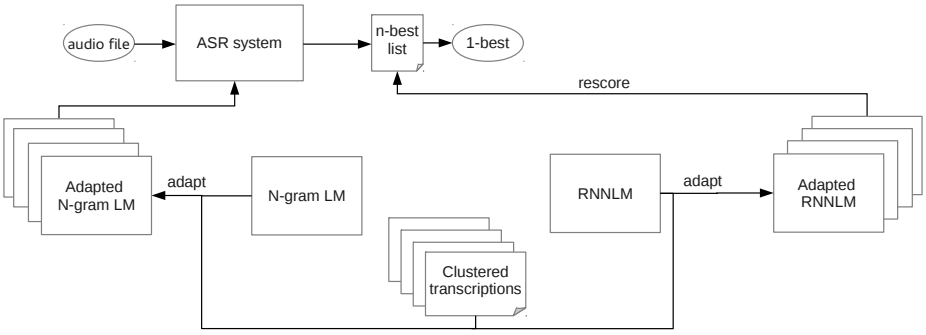


Fig. 1. Overview: language model adaptation to Code-Switching attitudes

The remainder of the paper is organized as follows: Section 2 reports on previous research in the area of Code-Switching, text clustering and language modeling using recurrent neural networks. Section 3 describes the SEAME corpus and analyzes it with focus on Part-of-speech tags triggering CS events. In section 4, we present how speakers can be clustered using the results of our analysis. Furthermore, we describe the adaptation of N-Gram and recurrent neural network language modeling for each Code-Switching attitude. In section 5, we present our experiments and results. The study is concluded in section 6.

2 Related Work

For this work, three different topics are investigated: 1) analysis of CS points and their integration into language models, 2) text clustering using similarity measures and 3) recurrent neural network language modeling and adaptation to more specific data. This section gives a short overview of prior work in these fields.

2.1 The Code-Switching Phenomenon

In [4, 12, 13], it is observed that code switches occur at positions in an utterance where they do not violate the syntactical rules of the involved languages. On the one hand, Code-Switching can be regarded as a speaker dependent phenomenon [3]. On the other hand, particular CS patterns are shared across speakers [14]. It is shown that speakers mainly switch to another language for nouns and object noun phrases. Therefore, the most frequent switches are between determiners and nouns and between verb phrases and object noun phrases.
In [16], different machine learning algorithms (for instance the Naive Bayes Classifier) trained on textual features are used to predict CS points. As features, word form, language identity, Part-of-Speech tags and the position of the word relative to the phrase are used. [5] compares four different kinds of N-Gram language models to predict Code-Switching. It is discovered that a class-based model which clusters all foreign words into their POS classes achieves the best performance. In [1], we show that the integration of POS tags into a neural network, which predicts the next language as well as the next word, leads to significant reductions in terms of perplexity.

2.2 Clustering Textual Documents

There are different text clustering techniques, such as hierarchical clustering (bottom-up or top-down) or k-means. While hierarchical clustering often provides better results, its time complexity is quadratic. On the other hand, k-means has a linear time complexity. Each technique requires a distance or similarity measure. The most common measure is the cosine measure [17].

2.3 Recurrent Neural Networks and their Adaptation

In the last years, neural networks have been used for a variety of tasks. [9] introduces a refined form of neural networks for the task of language modeling. The so-called recurrent neural networks are able to handle long-term contexts since the input vector does not only contain the current word but also the previous output from the neurons in the hidden layer. It is shown that these networks outperform traditional language models, such as N-Grams which only contain very limited histories. In [10], the network is extended by factorizing the output layer into classes to accelerate the training and testing processes. Recently, further information has been added to neural networks. [15] augments the input layer to model features, such as topic information or Part-of-Speech tags. In [6], it is shown that adaptation of Recurrent Neural Network Language Models in form of one-iteration retraining leads to improvements in the word error rate when the adapted models are applied for rescoring.

3 Code-Switching Prediction Using Part-Of-Speech

This section describes the SEAME data corpus and the analyses which we performed on the data: 1) A speaker independent analysis in which we compute

the CS rate after each Part-of-speech tag over all speakers and 2) A speaker dependent analysis in which the CS rate per speaker is calculated.

3.1 SEAME Corpus

SEAME (South East Asia Mandarin-English) is a conversational Mandarin-English Code-Switching speech corpus recorded from Singaporean and Malaysian speakers, created and collected by [7]. The corpus was used for the research project 'Code-Switch', jointly performed by Nanyang Technological University (NTU) and Karlsruhe Institute of Technology (KIT). The recordings consist of spontanously spoken interviews and conversations of about 63 hours of audio data. For this task, all hesitations are deleted and the transcribed words are divided into four categories: English words, Mandarin words, particles (Singaporean and Malaysian discourse particles) and others (other languages). These categories are used as language information in our neural networks. The average number of code switches between Mandarin and English is 2.6 per utterance. The duration of monolingual segments is very short: More than 82% English and 73% Mandarin segments last less than 1 second with an average duration of English and Mandarin segments of only 0.67 seconds and 0.81 seconds respectively. In total, the corpus contains 9,210 unique English and 7,471 unique Mandarin vocabulary words. The corpus is divided into three disjoint sets (training, development and test set). The data is assigned to them based on several criteria (gender, speaking style, ratio of Singaporean and Malaysian speakers, ratio of the four categories, and the duration in each set). Table 1 lists the statistics of the SEAME corpus in these sets.

Table 1. Statistics of the SEAME corpus

	Train set	Dev set	Eval set
# Speakers	139	8	8
Duration(hours)	59.2	2.1	1.5
# Utterances	48,040	1,943	1,018
# Token	525,168	23,776	11,294

3.2 Assigning POS Tags to Code-Switching Data

To be able to assign Part-of-Speech tags to our bilingual text corpus, we use two different taggers: On the one hand, the Stanford log-linear POS tagger for Mandarin and on the other hand, the Stanford log-linear POS tagger for English [19, 20]. The tags are derived from the Penn Treebank POS tag set for Mandarin and English [8, 22]. First, we determine Mandarin as matrix language (the main language of an utterance) and English as embedded language. If three or more words of the embedded language are detected, they are passed to the English

tagger. The rest of the text is passed to the Chinese tagger, even if it contains foreign words. The idea is to provide the tagger as much context as possible. However, most English words in the Mandarin segments are falsely tagged as nouns by the Chinese tagger. To avoid subsequent errors in the determination of trigger POS tags, we add a postprocessing step to the tagging process: We select all foreign words in the Mandarin segments and pass them to the English tagger in order to replace the wrong tags with the correct ones.

3.3 Speaker Independent Analysis

After having tagged the CS text, we select those tags that possibly predict CS points. The results are shown in table 2. First, we consider only those tags that appear in front of a CS point from Mandarin to English. Second, we investigate the tags predicting a CS point from English to Mandarin. In each case, only those tags are counted that occur more than 250 times in the text. Table 2 shows that CS points are most often triggered by determiners in Mandarin and by verbs and nouns in English. This seems reasonable since it is possible that a Mandarin speaker switches for the noun to English and afterwards back to Mandarin. It also corresponds to previous investigations as described in section 2.

Table 2. Mandarin and English POS that trigger a CS point

Tag	meaning	frequency	CS-rate
DT	determiner	11276	40.44%
DEG	associative 的	4395	36.91%
MSP	other particle	507	32-74%
VC	是	6183	25.85%
DEC	的 in a relative-clause	5763	23.86%
NN	noun	49060	49.07%
NNS	noun (plural)	4613	40.82%
RP	particle	330	36.06%
RB	adverb	21096	31.84%
JJ	adjective	10856	26.48%

3.4 Speaker Dependent Analysis

The previous analysis detects CS rates up to less than 50%. Thus, the triggering may not be reliable. A possible reason is that one speaker switches often after a specific tag while other speakers do not. Hence, a speaker dependent analysis is performed. The CS rate for each tag is computed for each speaker. Then, minimal, maximal and mean values and standard deviations are calculated. Indeed, the spread between minimal and maximal values is quite high for most of the tags. Figure 2 shows the distribution of the speaker dependent CS rates for all tags that appear more than 250 times in the text.

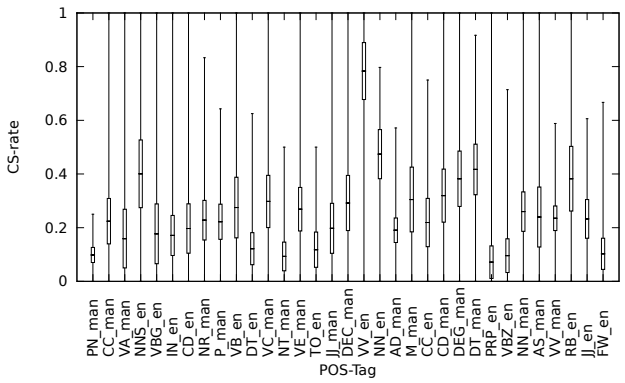


Fig. 2. Distribution of speaker dependent CS rates

To sum up, whether a Part-of-speech tag triggers a CS event is rather speaker dependent. This corresponds to the previous investigations described in section 2. Hence, a model that includes all individual deviations cannot be very precise.

4 Code-Switching Attitude Dependent Language Modeling

As shown in section 3, Code-Switching attitude is speaker dependent. Hence, we perform a clustering of the manual transcriptions of all speakers of our training data into K different groups to describe different Code-Switching attitudes. After that, we are able to adapt our language model to each class. Thus, we obtain K different language models that model Code-Switching more precisely and, therefore, achieve better recognition results.

4.1 Text Clustering

We apply the k-means algorithm to cluster the training transcriptions. As similarity measure, we chose the cosine distance because it was applied succesfully to cluster documents in the past. The following equation shows the computation of the cosine distance. d_1 denotes a vector representing document 1 and d_2 a vector for document 2.

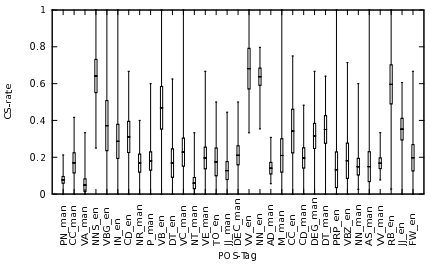
$$Dist(d_1, d_2) = (d_1.d_2)/(||d_1|| \cdot ||d_2||) \tag{1}$$

For the Code-Switching modeling, we define the document vectors d as follows:

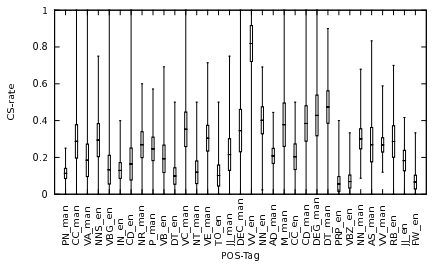
$$d = [f_{cs}(POS_1)/f(POS_1), ..., f_{cs}(POS_n)/f(POS_n)] \tag{2}$$

$f_{cs}(POS_i)$ defines the number of switches after the Part-of-Speech tag i in the given document while $f(POS_i)$ refers to the number of all occurences of the

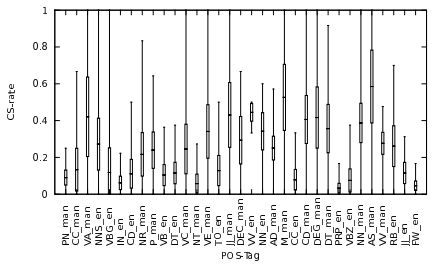
tag. After the clustering process converges (when there are no changes in the clusters), we use the mean vector of each cluster as representant.
Figure 3 shows for the example of three classes that clustering helps to decrease the spread of the Code-Switching attitudes. There are still tags for which the clustered speakers show different attitudes but there are also tags for which their attitude is quite similar. For example, the spread of the English tag 'NN' (noun) is discriminated into upper and lower values by the classes.



(a) distribution in class 1



(b) distribution in class 2



(c) distribution in class 3

Fig. 3. Distribution of speaker dependent Code-Switching rates after clustering

Further analyses show that, on the one hand, the classes divide different nationalities while, on the other hand, the gender of the speakers and the speaking style is similar in all classes. Hence, Code-Switching attitude seems to depend on the nationality but not on the gender or style. Table 3 summarizes the most important results for three classes.

Table 3. Analysis of the speakers that are clustered into one class

(con. denotes conversational speech, while iv. stands for interview)

Class	nationalities	gender	style
1	66 % Malaysia, 34 % Singapour	58 % female, 52 % male	5 % con., 95 % iv.
2	7 % Malaysia, 93 % Singapour	55 % female, 45 % male	47 % con., 53 % iv.
3	0 % Malaysia, 100 % Singapour	66 % female, 34 % male	29 % con., 71 % iv.

4.2 Language Modeling

Training We train two different language models (LM): An N-Gram LM for the decoding process and a recurrent neural network LM for rescoring.

N-Gram Language Model for Code-Switching We use the SRI Language Modeling Toolkit [18] to build trigram LMs from the SEAME training transcriptions. These models are interpolated with two monolingual language models that were created from 350k English sentences from NIST and 400k Mandarin sentences from the GALE project. The vocabulary of 30k entries contains all words in the transcriptions and the most frequent words in the monolingual corpora. Furthermore, characteristics of Code-Switching from the SEAME training transcriptions are analyzed and additional Code-Switching text is generated artificially as described in [21].

Recurrent Neural Network Language Modeling for Code-Switching In this paragraph, we describe the original version of the RNNLM toolkit [11] and our extension to it which is illustrated in figure 4. Vector $w(t)$, which represents the current word using 1-of-N coding, forms the input of the RNN. Its dimension equals the size of the vocabulary. Vector $s(t)$ contains the state of the network and is called 'hidden layer'. The network is trained using back-propagation through time (BPTT), an extension of the back-propagation algorithm for RNNs. With BPTT, the error is propagated through recurrent connections back in time for a specific number of time steps. Hence, the network is able to remember information for several time steps. The matrices U, V and W contain the weights for the connections between the layers. They are learned during the training phase. Moreover, the output layer is factorized into classes to accelerate the training and testing processes. Every word belongs to exactly one class. The classes are formed during the training phase depending on the frequencies of the words. Vector $c(t)$ provides the probabilities for each class and vector $y(t)$ the probabilities for each word given its class. Hence, the probability $P(w_i|history)$ is computed as shown in equation 3.

$$P(w_i|history) = P(c_i|s(t))P(w_i|c_i, s(t)) \tag{3}$$

In our extension of the RNNLM, the output classes do not depend on word frequencies but on languages. We use the language categorization described in section 3.1. Therefore, our model consists of four classes: One class for English words, one for Mandarin words, one for other languages and one for particles. We do not only intend to predict the next word but also the next language in our bilingual corpus. Hence according to equation 3, the probability of the next language is computed first and then the probability of each word given the language. Furthermore, we add another vector $f(t)$ to the network which provides features corresponding to the current word and concantenate the former input layer with this vector. According to the analyses described in section 3, we use POS tags as features. Vector $f(t)$ consists of 67 elements (31 Mandarin POS tags, 34 English POS tags, one feature for words classified as other languages

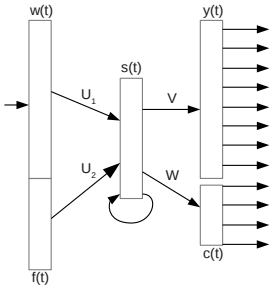


Fig. 4. RNNLM for Code-Switching
based upon a figure in [10]

and one feature for particles). During the training and testing phases, not only the current word is activated but also its feature. Because the POS tags are integrated into the input layer, they are also propagated into the hidden layer $s(t)$. Thus, features from several previous time steps are stored in the history. Hence in equation 3, the term $P(c_i|s(t))$ computes the next language c_i using not only information about previous words, but also previous features.

Adaptation After the clustering, we retrain our models with the data of the different clusters. For the N-Gram model, we interpolate the baseline N-Gram model with an N-Gram model trained on the texts of one class. Hence, we obtain one N-Gram model per class. The interpolation weights are chosen to minimize the perplexity of the development set speakers that are similar to the class. Analog to this, we retrain one RNNLM per class. We perform one-iteration training using the texts of the different classes.

Rescoring The decoding process contains two different passes. In the first pass, we run the speech recognition system to extract the N-best hypotheses using the speaker independent N-Gram LM. Based on the average score of the CS attitude dependent RNNLM on these N-bests, each speaker is assigned to a specific CS attitude. In the second pass, we rerun the decoding process using our CS attitude dependent N-Gram for each speaker and rescore the 100-best hypotheses using the CS attitude dependent RNNLM to obtain the best hypothesis.

5 Experiments and Results

This section reports the experiments and evaluations performed on the challenge of CS language modeling. Since the models are adapted to fit better to individual speakers, their perplexities are computed speaker-wise.

5.1 Clustering Experiments with K-Means

The most important parameter in the clustering process is the cluster size. Hence, different sizes are tested. Since rescoring experiments with the RNNLM are faster than decoding experiments with the N-Gram model, the following values are calculated using the 100-best lists of the speaker independent system and the RNNLM system to compute perplexities and rescore the hypotheses.
Table 4 summons the minimum and maximum perplexity on the eight development set speakers in order to detect the most appropriate cluster size.

Table 4. Minimum and maximum perplexity on the development set speakers

Speaker	Baseline	2 classes	3 classes	4 classes	5 classes
Speaker 1	257.47	234.29 - 270.57	234.08 - 270.56	233.39 - 267.56	237.32 - 274.98
Speaker 2	221.00	194.78 - 218.96	194.66 - 219.04	194.41 - 216.52	196.96 - 222.16
Speaker 3	253.31	242.94 - 283.21	243.54 - 283.44	242.87 - 280.27	242.03 - 289.04
Speaker 4	201.28	186.14 - 213.38	186.70 - 213.55	185.96 - 212.29	188.37 - 217.14
Speaker 5	339.50	299.69 - 355.34	299.84 - 355.75	299.58 - 349.79	303.15 - 366.97
Speaker 6	151.92	135.00 - 156.76	135.05 - 156.81	134.92 - 156.67	135.49 - 160.82
Speaker 7	225.82	221.99 - 251.83	222.00 - 250.66	223.56 - 252.66	220.47 - 279.62
Speaker 8	194.35	189.30 - 206.97	189.30 - 206.32	188.97 - 207.64	191.10 - 222.73

It can be noted that the results of two, three and four classes are quite similar and superior to a cluster size of five. Although the worst result per cluster performs worse than the baseline model, most of the classes of each cluster lead to an improvement of the perplexity. These results support the speaker dependent analysis: It is possible to adapt the language model to individual Code-Switching attitudes. The three best cluster sizes (2, 3, and 4 classes) are further evaluated regarding their word error rate reduction in the rescoring process. This results in a best cluster size of 3 classes. This is reasonable since two classes might not cover enough different speaker attitudes, while four or more classes do not contain enough training data per class. Hence, a cluster size of three is chosen for further evaluations.

5.2 Results

This subsection summons the results of our experiments, including the test of our models on the evaluation set speakers. Table 5 shows the minimum and maximum perplexities per speaker of all three adapted models and the baseline model for each the N-Gram and the RNN LM. Again, the models adapted on the clustered training data can improve the performance in terms of perplexity for all speakers. Finally, the adapted models are used to decode and rescore the evaluation set speakers. We use the system in [21] to perform the decoding process. As performance measure, the Mixed Error Rate (MER) as proposed in [21] is calculated. It applies word error rates to English and character error

Table 5. Minimum and maximum perplexities on the evaluation set speakers

Speaker	N-Gram	Adapted N-Gram	RNNLM	Adapted RNNLM
Speaker 1	317.84	302.94 - 326.24	200.66	197.74 - 204.82
Speaker 2	265.77	253.73 - 270.81	181.60	175.85 - 185.48
Speaker 3	327.09	302.56 - 352.60	187.04	170.92 - 197.30
Speaker 4	232.83	213.33 - 248.30	174.13	160.58 - 185.28
Speaker 5	367.72	365.47 - 409.25	364.59	327.33 - 392.68
Speaker 6	175.28	162.02 - 181.83	275.89	253.67 - 299.37
Speaker 7	318.50	306.58 - 375.41	286.31	286.30 - 292.29
Speaker 8	292.53	281.57 - 331.04	256.99	241.69 - 268.23

rates to Mandarin and is the weighted average over all English and Mandarin parts of the speech recognition output. By applying character based error rates to Mandarin, the performance does not depend on the applied word segmentation algorithm for Mandarin and thus the performance can be compared across different segmentations. Table 6 shows the results on the SEAME development and evaluation set.

Table 6. Mixed error rate results after decoding and rescoring with the adapted models

Model	development set	evaluation set
Speaker-independent N-Gram model	34.74%	29.23%
Adapted N-Gram model + RNNLM	34.47%	28.89%

6 Conclusions

In this paper, we showed that Code-Switching is a speaker dependent phenomenon. Therefore, we clustered similar Code-Switching attitudes using cosine-distances. Furthermore, we trained recurrent neural network language models for the Code-Switching task by adding POS information to the input layer and by factorizing the output layer into languages. Afterwards, we adapted our background N-Gram and RNN language model using the corresponding training texts of these clusters. We showed that this approach leads to significant reductions in terms of perplexity. Finally, we used these adapted language models to rerun and rescore the speech recognition system for each speaker and achieved some improvements in terms of mixed error rate.

References

1. Adel, H., Vu, N.T., Kraus, F., Schlippe, T., Schultz, T., Li, H.: Recurrent neural network language modeling for code switching conversational speech. In: ICASSP (2012)

2. Auer, P.: Code-switching in conversation. Routledge (1999)
3. Auer, P.: From codeswitching via language mixing to fused lects toward a dynamic typology of bilingual speech. International Journal of Bilingualism 3(4), 309–332 (1999)
4. Bokamba, E.G.: Are there syntactic constraints on code-mixing? World Englishes 8(3), 277–292 (1989)
5. Chan, J.Y.C., Ching, P., Lee, T., Cao, H.: Automatic speech recognition of cantonese-english code-mixing utterances. In: Proc. of ICSLP (2006)
6. Kombrink, S., Mikolov, T., Karafiát, M., Burget, L.: Recurrent neural network based language modeling in meeting recognition. Proc. of ICSLP (2011)
7. Lyu, D.C., Tan, T.P., Chng, E.S., Li, H.: An analysis of a mandarin-english code-switching speech corpus: Seame. ICSLP (2010)
8. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Computational linguistics 19(2), 313–330 (1993)
9. Mikolov, T., Karafiát, M., Burget, L., Cernocky, J., Khudanpur, S.: Recurrent neural network based language model. In: Proc. of ICSLP (2010)
10. Mikolov, T., Kombrink, S., Burget, L., Cernocky, J., Khudanpur, S.: Extensions of recurrent neural network language model. In: ICASSP. pp. 5528–5531 (2011)
11. Mikolov, T., Kombrink, S., Deoras, A., Burget, L., Cernockỳ, J.: Rnnlm–recurrent neural network language modeling toolkit. In: Proc. of the 2011 ASRU Workshop. pp. 196–201 (2011)
12. Muysken, P.: Bilingual speech: A typology of code-mixing, vol. 11. Cambridge University Press (2000)
13. Poplack, S.: Syntactic structure and social function of code-switching, vol. 2. Centro de Estudios Puertorriqueños,[City University of New York] (1978)
14. Poplack, S.: Sometimes i'll start a sentence in spanish y termino en español: toward a typology of code-switching. Linguistics 18(7-8), 581–618 (1980)
15. Shi, Y., Wiggers, P., Jonker, C.M.: Towards recurrent neural networks language models with linguistic and contextual features. ICSLP (2012)
16. Solorio, T., Liu, Y.: Learning to predict code-switching points. In: Proc. of the EMNLP. pp. 973–981. Association for Computational Linguistics (2008)
17. Steinbach, M., Karypis, G., Kumar, V., et al.: A comparison of document clustering techniques. In: KDD workshop on text mining. vol. 400, pp. 525–526 (2000)
18. Stolcke, A., et al.: Srilm-an extensible language modeling toolkit. In: Proc. of ICSLP. vol. 2, pp. 901–904 (2002)
19. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proc. of the HLT-NAACL. pp. 173–180 (2003)
20. Toutanova, K., Manning, C.D.: Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In: Proc. of the 2000 Joint SIGDAT conference EMNLP/VLC. pp. 63–70 (2000)
21. Vu, N.T., Lyu, D.C., Weiner, J., Telaar, D., Schlippe, T., Blaicher, F., Chng, E., Schultz, T., Li, H.: A first speech recognition system for mandarin-english code-switch conversational speech. In: ICASSP. pp. 4889–4892 (2012)
22. Xue, N., Xia, F., Chiou, F.D., Palmer, M.: The penn chinese treebank: Phrase structure annotation of a large corpus. Natural Language Engineering 11(2), 207 (2005)