Language Modeling for Voice-Enabled Social TV Using Tweets

Junlan Feng, Bernard Renger
AT&T Labs-Research, 180 Park Avenue, Florham Park, NJ 07932, USA
junlan@research.att.com, renger@research.att.com

Abstract

Social TV is a recent trend that integrates social media access and TV viewing. In this paper, we investigate approaches for building effective language models for a voice-enabled social TV application, where viewers can speak their social media updates while watching TV. We propose to take advantage of social media data, more specifically TV-related Twitter messages (tweets). The challenge is the noisy nature of Twitter data. Our contributions are as follows. First, we collect TV show related tweets and provide a detailed analysis of the style mismatch between written tweets and spoken language. Second, we propose a learning based approach for transforming tweets to be more suitable for language modeling. This transformation considers lexical, phonetic and contextual similarity between the misspellings and the canonical form. Third, we build the language models from normalized TV-related tweets along with other data resources that are weighted to optimize speech recognition performance. The model created via normalized tweets achieved higher performance.

Index Terms: Voice-Enabled Social TV, Text Normalization, Language Modeling

1. Introduction

The phenomenal rise of social networks in the past few years is changing the way viewers perceive their TV watching experience. According to a Nielsen report, 57% of Internet users use TV and Internet simultaneously at home. Cable providers, software companies, and hardware manufacturers are launching products with social TV functionalities, such as AT&T U-verse, Verizon FiOS, Apple TV, Yahoo! Connected TV, Google TV, Boxee, etc. Most of these applications allow users to read social media messages on TV. In [1], we presented a voice-enabled social TV system, which allows users to interact, follow and monitor on-line social discussions related to a TV show while watching TV. To make the interaction easier, we use automatic speech recognition (ASR) to enable users to speak their messages.

In this paper, we explore the problem of language modeling (LM) for higher ASR accuracy in social TV. We propose using TV-related on-line social media posts along with other data resources to train the language models. The challenges lie in the noisy nature of social media data. Typos, abbreviations, intentional and unintentional misspellings, and emotion expressions are all mixed in. We choose to use tweets, namely short messages on Twitter, since tweets are more real-time and there is a large amount of data available. The remainder of the paper is organized as follows. In Section 2, we describe our process for collecting TV-related tweets and provide an analysis of the obtained tweets. Section 3 presents our approach for normalizing the tweet text. In Section 4, we adapt the language models for the social TV application. In Section 5, we present our experiments. We summarize our findings and future work in Section 6.

2. Tweet Filtering and Analysis

2.1. Tweet Filtering for TV Shows

It is not trivial to monitor a tweet stream and select those relevant for any given TV show. Some shows have very ambiguous titles such as house, which will lead to low precision when simply using titles as queries. The shows with a long title, on the other hand, often result in low recall. In [2], we described a machine learning based bootstrapping approach for collecting relevant tweets for a given TV show. First, we started with a small set of annotated data, where for a given show and a candidate message, we annotated the pair to be relevant or irrelevant. From this annotated data set, we trained an initial classifier. The features are designed to capture the association between the TV program and the message. Second, using our initial classifier and a large data set of unlabeled messages, we derived broader features for a second classifier to further improve precision. Additionally, some of our obtained features are also used to improve the recall of the system. Details of these features can be found in [2]. With this approach, we obtained a classifier performing at 84.3% precision.

In this paper, for our Twitter data set, we randomly selected three million TV-related tweets filtered by this classifier between September 2011 and March 2012. Table 1 provides a few examples of TV show related tweets from our collection.
cool spongebob is the show! lol! Im watchin say yesssss to the dress. too l8
RT @codysimpson: can’t wait to perform on the #TheEllenShow. http://t.co/UPLuoJV #coastocoast
@EllapageMusic no joke i actually adore Smallville! aaaaagh

Table 1: Tweet Examples

2.2. Challenges for Language Modeling Using Tweets

In general, N-gram language modeling for ASR requires a large amount of in-domain data that matches the topics and the style for users that speak their tweets while watching TV shows. TV show related textual tweets are a sensible substitute for real in-domain data since they fit the domain well in terms of topic and style. We analyze our collected data in this section to highlight the challenges.

First is the special entities in tweets. RT indicates a tweet is a retweet, a forwarded message from a previous one. @USERNAME indicates a mention of a specific user or a reply to the defined user. The symbol # indicates a hashtag, which is often added by the user as a topic tag for the tweet. Users also use hashtags as regular words in the text. Another common entity in tweets is URLs. It is challenging for ASR to recognize these entities, since many of them naturally are hard to pronounce and the vocabulary is dynamically changing. In our data collection, 18.8% of the tweets include hashtags; 48.1% contain a URL; 40.5% mention a Username; and 17.0% are retweets starting with RT.

Second is intentional misspellings such as oooook for ok and exciiiiited for excited. Users on Twitter tend to intentionally misspell some words to express their emotions. Sometimes a misspelled word is considered more fashionable such as using watchin a show instead of watching a show. This results in an unnecessarily large vocabulary. Creating reasonable pronunciations for misspelled words is a challenge for ASR.

Third is unintentional misspellings (typos) such as minesota, which is similar to the noisy nature of query logs. They usually look like or sound like the canonical form.

Fourth is abbreviations. Various abbreviations are frequently used in tweets to shorten the message such as omg for Oh My God and 2nite for tonight. Many of them are commonly known by Twitter users. New abbreviations are also invented frequently.

In our collection of three million tweets, there are 42 million tokens and around 370 thousand unique words. A high percentage (75.1%) of these words are out-of-vocabulary (OOV). We used a set of dictionaries as references. More details of these dictionaries are described in Section 3.3.

3. Normalizing Tweets

In this section, we describe related work and our approaches for normalizing TV-related tweets.

3.1. Related Work

There have been a few approaches proposed recently to normalize tweets for natural language processing and text mining. In general, the goal of normalization is to convert misspelled words into the corresponding canonical form and to convert non-grammatical sentences into grammatical sentences. The first line of approach is borrowed from general text normalization based on lexical similarity between the misspelled word and the correct spelling. For instance, according to edit distance watchin is similar to watching. There are several alternative string similarity metrics [3]. The second approach examines phonetic similarity [4] [5]. It basically checks if the misspelled word sounds like a word in the vocabulary. IS and late in this case are close. The third approach looks into expanding abbreviations in tweets. Because each tweet is limited to 140 characters, users tend to use more abbreviations in tweets, such as lol for Laugh Out Loud. The fourth approach considers the contextual similarity of the misspelled word and the intended word. We incorporate some of these techniques in our work and propose a learning based approach which considers all the factors mentioned above and beyond. Details are below.

3.2. Phase-I Normalization

Though Twitter entities such as hashtags and URLs are key features of tweets as discussed in Section 2, we observe from the spoken tweets we have that users who post tweets by speaking to mobile applications tend to avoid using these entities. It could be because these entities in general are hard to speak and hard to remember, or users may have experienced that ASR fails to recognize them when they try. Given this observation, in Phase-I, we clean up tweet text by removing Usernames, Hashtags and URLs.

3.3. Phase-II Normalization

In Phase-II, we focused on correcting misspelled words in tweets. Table 2 summarizes our proposed Phase-II normalization process. It takes as input the original tweets and a few dictionaries. At Step 1, it detects OOV words by comparing to the dictionaries. It then detects and corrects intentional OOV words at Step 2. Step 3 applies a machine learning algorithm to correct misspelled words. Step 4 handles abbreviations. OOV Word Extraction: In step 1, we extract OOVs word by comparing to a set of dictionaries. The set includes Aspell General English Dictionary1, a

1http://aspell.net/
Input:
A Collection of Tweets & Dictionaries & Gazetteers

Normalization:
Step 1: Extract OOV words by comparing to the dictionaries
Step 2: Detect intentional OOV words and shorten them to the possible canonical form
Step 3: Correct the misspelled words
Step 4: Detect Abbreviations and change them to their spoken form

Table 2: Normalization Procedure

SMS/Twitter Abbreviation Dictionary², Yellow Pages business location dictionary³, and an English People Name dictionary, as references.

Intentional OOV Word Correction: The usual rules of English spelling outlaw triple or more repeated letters. On Twitter, users tend to use words like yesssss and aaaaaaaaawww to express users’ emotion and emphasis on certain words. However, these misspellings rarely correspond to spoken words. We examined our OOV words and replaced a sequence of 3 or more repeated letters with 2 letters. For instance, both yessssss and yessss become yess. In our OOV word list, we find around 10.0% of the OOV words fall into this category. If this corrected word is still an OOV word, we further try reducing the 2-letter repeats to one letter to see if it becomes an in-vocabulary word such as reducing yess to yes. If not, we keep the 2-letter correction. This step sometimes won’t correct the word completely but it brings the spelling closer to the canonical form.

Correcting Misspelled Words: We consider lexical similarity and phonetic similarity to choose candidate corrections for a misspelled word. Once the candidate set is obtained, a classifier using a broader set of features is used to rank the candidates. First, we generate the phoneme sequence for each word in the vocabulary and each OOV word using a TTS system, which performs the grapheme-to-phoneme conversion by using a large dictionary and backup rules. The specific system we used is AT&T Natural Voices⁴. We apply Lucene ⁵, a text search engine, to index the words and their phoneme sequences. Second, for each OOV word oov in a tweet t, we search for the similar words w in the vocabulary in terms of similarities to the letter sequence and phoneme sequence using Lucene fuzzy search. Lucene fuzzy search is built upon Edit Distance. We use these words as the candidates for the correction. Third, we train and use a classifier to determine if the candidate word w is the right correction for the given oov in the tweet t. We randomly select 200 OOV words in 1000 tweets and annotate these OOV words by their true spelling. We use this data set as our training data for the classifier. Each pair of oov and w given the tweet t is a training and testing unit. We used RandomForest as our training algorithm [6]. The candidate classified as True with the highest score is output as the correction. We achieved 92% classification accuracy on 10-fold cross-validation. This classifier is used to choose the final correction for any given OOV word in tweets. After we adjust the threshold on the classifier confidence score, we obtained 81.0% precision in selecting the correct candidate. With this setup, we are able to reduce the vocabulary size of this data set by 51%. Features we used for classification include lexical similarity, phoneme sequence similarity, contextual similarity, whether oov is capitalized, whether the first letters of oov and w match, the character N-grams where they differ (e.g., “azu” vs. “asu” for pleasure and pleasure), as well as the relative frequency of the candidate correct word. We measure contextual similarity by calculating the percentage of overlap words between t and the tweets containing w in the data set.

Abbreviations: We use the SMS dictionary, which has 900 entries, to check abbreviations in our data. We found that 308 out of the 108K OOV words in our data set are abbreviations. However, they account for 7.4% of OOV usage. We decided to keep their original form and make them all upper case in the tweet text so that the ASR pronunciation model would treat them as abbreviations instead of regular words. For instance, LOL will be considered as a letter sequence L O L instead of the word lol.

4. Language Model

We use interpolation to combine several models, which are estimated on different sources. In general, interpolation learns interpolation weights by minimizing the language model perplexity on a held-out in-domain development data set. We have trained a general language model using broadcast news, web data, and general spoken SMS messages. In order to adapt this model to the social TV application, we interpolate it with TV-related data, in our case, TV-related tweets. Details of our experiments are given in the experiment section.

5. Experiment

In this section, we describe the data sets we used for training and testing and the experimental results in terms of ASR word accuracy (WACC). The AT&T Watson (SM) ASR engine [7] was used to create the LMs.

5.1. Data Sets

We used three data sets for training. Data Set I is a mix of multiple resources including broadcast news (1.2 million sentences), question/answering queries (10.1 million sentences), spoken SMS messages (1.4 million sentences)
from a deployed mobile phone application, typed SMS messages (5.0 million sentences), and web search queries (9.3 million sentences). A general baseline LM (General-LM) was created from this data set. Data Set II contains three million TV show related tweets cleaned up by our Phase-I normalization process. These tweets were selected as described in Section 2. Data Set III contains the same three million TV-related tweets but cleaned up by our Phase-II normalization process. The results of the Phase-I normalization were used as input for the Phase-II normalization.

To collect TV show related spoken tweets for evaluation, we created an iPhone/iPad application, where a user can pick a show from a list of 32 popular shows and see the show title, cast information, and the cover art picture. Users are encouraged to speak tweets about the selected show by pressing the click to speak tweet button but they are not limited to only tweeting about these shows. Many spoken tweets we collected are about other shows the user is familiar with. With this application, we obtained 483 utterances, which were then transcribed and randomly divided into two sets: the TV-Test test set (237 utterances) and the TV-Dev set (246 utterances). We used this data set collected from an iPhone/iPad application instead of utterances spoken to TV or a remote control because it’s easy to implement. Another consideration is that users are already watching TV on their mobile devices or using mobile devices as a companion device.

5.2. Results

We trained a Katz backoff 3-gram baseline LM (General-LM) using Data Set I. We created the second LM (TV-PhaseI-LM) by interpolating a LM we estimated from Data Set II and the General-LM. The third LM (TV-PhaseII-LM) was built by interpolating the LM built on Data Set III and the General-LM. TV-Dev was used as the development data set for interpolation. ASR word accuracies using these three LMs with the test set TV-Test are reported in Table 3. The word accuracy with General-LM is 60.1%. The performance improves to 63.6% using TV-PhaseI-LM. The performance improves slightly to 63.8% using TV-PhaseII-LM. Though it is not encouraging that TV-PhaseII-LM doesn’t improve the performance significantly, it is partially explainable. In text tweets, users often commented on episodes, actors, characters and expressed their opinions, where abbreviations and misspellings typically occur. In TV-Test, very few utterances contain information with the same level of detail. Even abbreviations such as lol which frequently occur in TV-related tweets rarely appear in the TV-related test utterances. There are two possible reasons. One is that users may have tried speaking tweets about TV shows as they typically do on Twitter via text and experienced difficulty. Hence, they shy away from such attempts. Secondly, to some extent, users feel they can’t easily utter a tweet that they typically type, especially when they need to express emotions using emoticons and need to emphasize certain words. Since TV programs are mainly for entertainment, it’s more likely that users need to express opinions about a TV show in such a style.

<table>
<thead>
<tr>
<th>LM</th>
<th>Test Set</th>
<th>WACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>General-LM</td>
<td>TV-Test</td>
<td>60.1%</td>
</tr>
<tr>
<td>TV-PhaseI-LM</td>
<td>TV-Test</td>
<td>63.6%</td>
</tr>
<tr>
<td>TV-PhaseII-LM</td>
<td>TV-Test</td>
<td>63.8%</td>
</tr>
</tbody>
</table>

Table 3: ASR Performance

6. SUMMARY

In this paper, we proposed to improve the language model for ASR in a social TV application using TV-related tweets. We described our approach for filtering tweets for TV shows and analyzing the challenges of using these tweets. We then developed a learning based procedure to correct intentional and unintentional misspellings with 81.0% accuracy. We adapted a general LM by interpolating it with a LM trained on our normalized TV tweets. With this new LM, we observed 3.7% absolute improvement on ASR word accuracy.

For future work, we will improve our normalization approach by exploring broader features for the classifier and using more training data. Collecting or finding a more representative test data set will also be necessary. Another interesting investigation is to find out how much improvement can be achieved using a random set of tweets instead of focusing on TV-related tweets. We are also interested in building show-specific language models using Twitter data.

7. ACKNOWLEDGMENTS

We are grateful to Mazin Gilbert, Behzad Shahraray, Diamantino Caseiro, and Barbara Hollister for their help in providing data and discussing ideas in this paper.

8. References