Paraphrase Features to Improve Natural Language Understanding

Xiaohu Liu[1], *Ruhi Sarikaya*[1], *Chris Brockett*[2], *Chris Quirk*[2], *William B. Dolan*[2]

[1]Microsoft Corporation, Bellevue, USA
[2]Microsoft Research, Redmond, USA
{derekliu,rusarika,chrisbkt,chrisq,billdol}@microsoft.com

Abstract

Natural language understanding (NLU) systems for speech applications require large quantities of annotated data. We investigate the use of a domain-independent machine-translation-based paraphrase system to improve performance without incurring the costs of obtaining additional annotated data in an NLU system. Our experimental system incorporates Support Vector Machine (SVM) domain and intent models to detect intents, and a conditional random field (CRF) model to identify semantic slots in a given query. Two approaches are compared. In the first, we retrain models using generated paraphrases to augment the original training set. In the second, we use paraphrases as supplementary features of the original queries in the SVM and CRF models. Experiments in four domains indicate that incorporating paraphrase yields useful performance gains, and that the feature-based approach provides more stable performance than synthetic augmentation of training data.

Index Terms: natural language understanding, paraphrase, conditional random fields, support vector machines

1. Introduction

Conventional natural language understanding (NLU) systems automatically identify the domain and intent of natural language queries and extract semantic slots from the queries [1]. In a supervised machine learning approach using annotated training data, a system might typically use classifiers, such as support vector machine (SVM) models [2] for domain and intent detection, and taggers such as conditional random field (CRF) models [3] for slot extraction. When a user issues a new query, domain and intent classifiers first predict the domain of the query and the user intent conveyed by the query; taggers then extract semantic slots associated with the intent.

One major obstacle to building NLU systems of this kind, however, is that high accuracy requires large volumes of annotated training data. Domain-specific queries must be collected; each query must be manually annotated with domain, intent and slot information. The effort to collect and tag large quantities of natural language data is both time-consuming and costly, in many cases prohibitively so.

One means of improving NLU models without relying on more training data is to include more input features. In this work, we investigate paraphrasing as a means of generating these features. For our purposes, we will define distinct word sequences (phrases, sentences, etc.) as paraphrases if they convey the same or almost the same information in some context. In NLU, paraphrasing can be treated as mapping to different wording while maintaining the interpretation of the speaker—in other words, as a monolingual translation task [4].

Paraphrase technologies are finding broad application in natural language processing tasks [5]. In machine translation, paraphrase has been used to mitigate the impact of Out-Of-Vocabulary (OOV) items when training on small data sets [6], and to automatically evaluate machine generated translations against human-authored ones that may use different phrasings [7]. In query answering systems, questions are often phrased differently than in documents that contain the answers; taking such variations into account using machine translation can improve system performance significantly [8]. In text summarization, where the most important sentences of the texts to be summarized are identified, it is important to avoid extracting sentences that redundantly convey the same information (that is, are paraphrases) [9]. Similarly, in natural language generation [10], paraphrasing may be used to avoid repetition and to produce alternative expressions that improve readability.

In this paper, we demonstrate that paraphrase technologies are also generally applicable to NLU models of the kind used in speech applications. We focus on two approaches employing whole sentence paraphrases generated using statistical machine translation (SMT) techniques [4] from queries in a training set. In the first approach, generated paraphrases are added to the original training set to create a much larger training corpus. In the second, paraphrases act as additional features attached to existing training queries. To the best of our knowledge, this represents the first comprehensive effort to apply paraphrases in both classification and sequence labelling tasks for NLU.

2. Natural language understanding system

Our NLU system takes natural language text as input, predicts the semantic meaning, domain, intent and slots, using three types of NLU models, as depicted in Figure 1.

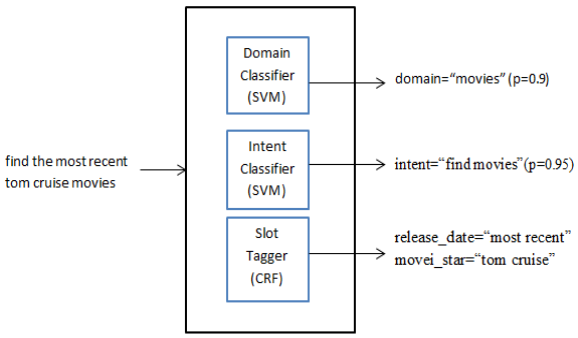

Figure 1: Natural language understanding system architecture

For each supported domain, we collect a set of in-domain queries. Queries are then annotated manually using a predefined semantic representation. The semantic representation of a user query is a triple of domain, intent, and slot list.

⟨DOMAIN, INTENT, SLOT_LIST⟩

A slot list is a list of key-value pairs:

⟨SLOT_NAME, SLOT_VALUE⟩

Each query belongs to one or multiple domains, such as MOVIES, MUSIC, APPLICATIONS, or GAMES. Also a query is classified into one of set of pre-defined domain-specific intents, such as FIND_MOVIES, PLAY_MOVIES, PURCHASE_SONGS, PLAY_SONGS, etc. Semantic slots associated with each intent specify detailed parameters for an intent.

For example, the semantic representation of the query "find the most recent tom cruise movies" is as follows:

⟨MOVIES, FIND_MOVIES, [⟨ MOVIE_STAR,"tom cruise" ⟩,
⟨ RELEASE_DATE,"most recent"⟩] ⟩

Multi-class support vector machine (SVM) models [2] are built for both domain and intent classifiers. Domain classifiers are used to identify the domains a query belongs to, and for each domain, a domain-specific intent model is constructed to distinguish intents within the domain. CRF models [3] are trained to label each word in a query with its semantic tag.

Feature sets used in SVM and CRF models include lexical ngrams to cover context words, the semantic tag of the previous word, and binary indicator features from dictionaries of named entities such as movie titles, movie actors, game titles, and application names [12]. In this paper, we attempt to use paraphrases as additional feature sets to improve all three models in our NLU system. This approach resembles the use of paraphrase models as features in Markov Random Fields for search [13] [14].

3. Paraphrase generation

Paraphrases are sets of distinct word sequences (phrases, sentences or longer sequences of text) that convey the same, or almost the same information. For example, sentences (1) and (2) are paraphrases, as they should produce the same effect.

(1) start internet explorer

(2) launch ie

Paraphrases generated using translation models built from query logs have been shown to improve search [13] [14]. In this paper, however, we generate paraphrases using a monolingual SMT system [4] built using data derived from multiple multilingual SMT systems. A phrasal decoder like that of the commonly-used Moses system [16] translates short English word sequences into different English word sequences. The translation model is trained with 350 million pairs of aligned parallel phrases, using the assumption that if two English phrases translate into the same foreign phrase, they are likely paraphrases of one another (cf [15]). In our system we draw on phrase tables for 17 languages paired with English as used in a large commercial machine translation engine. After filtering, the overall vocabulary size of the model is about 315K words in common current use. The paraphrase search algorithm uses these phrasal replacements in tandem with a very large general-domain English language model; the same language model is also used by bilingual translation systems in this commercial machine translation engine. This large-scale system generates replacement sentences in a broad range of domains. It also presents alignment information and a variety of scores that can be used to inform later processes. As with conventional bilingual statistical machine translation, the outputs are not always completely grammatical or lexically appropriate; we are to some extent able to offset this through the use of translation confidence scores returned by the translation engine.

4. Paraphrases to improve NLU models

The paraphrases generated by the above engine are used to extend our models. In other words, given an input query, we do not try to find individual paraphrases in our training set (as in some related approaches [8]). Instead, we compare two approaches to incorporating paraphrases into model building. In the first, the generated paraphrases are added to the original training set to form a much larger training set. In the second, we employ paraphrases as additional features of existing training queries.

4.1. Paraphrases as new records

In our first experimental setup, the paraphraser is used to synthesize additional queries in the hope of finding new variations. Using the original records from our in-domain query dataset as seeds, we run each query through the paraphraser to generate N-best paraphrases. The synthetic queries are annotated with the same domain, intent and slot tags as the input query, so as to obtain a larger training set with automatically annotated labels.

For example, sentence (2) "launch ie" is added to APPLICATIONS domain with intent START_APP since its paraphrase (1) is originally annotated as START_APP in APPLICATIONS domain; also, "ie" is tagged as APPLICATION_NAME as it is the aligned paraphrase of "internet explorer". In the examples below, sentence (3) is from the original training set in the MOVIES domain, and (4)-(8) are generated paraphrases.

(3) i would like to watch cartoons

(4) i would like to see comics

(5) i would like to see cartoons

(6) i would love to watch animation

(7) i want to see cartoons

(8) i would like to see animated

For the most part, these generated outputs are reasonably natural. We can potentially add (4) to (8) to the training set with the labels FIND_MOVIES as intent and MOVIES as domain in order to increase the training set coverage. The re-trained model from the new data set is expected to predict the right domain and intent labels for the new query "i want to see comics", even though words "see" and "comics" may not occur in the original training set.

When paraphrases are added to the training set, we also try to annotate semantic slots according to the original query annotation and word alignment. As the word "cartoons" in sentence (3) is manually annotated as MOVIE_GENRE, in all its paraphrases "comics", "animation" and "animated" are also assigned the same tag. Thus, the trained slot model contains richer contextual information about the slot MOVIE_GENRE. The added paraphrases contribute more weight to the words "see" and "watch" in predicting the following word as tag MOVIE_GENRE. Theoretically, the new CRF model is able to predict word "comedies" as MOVIE_GENRE with higher confidence score in the test query " i would love to see comedies".

Since our paraphraser generates N-best paraphrases with confidence scores, those outputs with very low confidence scores (e.g. sentence (9) above) are rejected to reduce noise from malformed outputs.

4.2. Paraphrases as features

As already noted, the generated paraphrases do not always constitute well-formed natural language sentences. Instead of adding paraphrases directly as new records, one alternative is to treat the generated outputs as features that enhance existing queries. In other words, sentences (4)-(8) are not added to the training set, but instead are associated with sentence (3) as a collection of features.

We take two different approaches to adding paraphrase features to SVM and CRF models. In SVM model training, we apply all words from paraphrases as features of the original query. For example, sentence (3) is augmented with the following features using all words in sentences (4)-(8):

{ i, would, like, to, see, comics, cartoons, love, watch, animation, want }

Words from the paraphrase word set are distinguished from the same word appearing in the original query so that the SVM model can weight them differently. For example, "cartoons" in the original query is treated as a different feature than "cartoons" from the paraphrases, and would be assigned a higher weight in predicting that the query belongs to the MOVIES domain.

The feature function of a paraphrase word w_i is defined as follows.

$$f_i(q_j, y_k) = \begin{cases} 1 & \text{if } q_j \in y_k \text{ and } w_i \in \text{para}(q_j) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

If there exists a paraphrase of the query q_j with label y_k contains word w_i, the function returns 1.

In CRF model training, words from paraphrases are associated with semantically similar words in the original queries. For example, from paraphrase (4) above, "see" is added as a feature associated with the word "watch" and "comics" as a feature of "cartoons".

Table 1 gives an example of the new training record with paraphrase features for sentence (3) for CRF model training. For space reasons, only the first four paraphrases (4) to (7) are shown. The first column is the original sentence, others are paraphrase features.

i	i	i	i	i
would	would	would	would	want
like	like	like	love	O
to	to	to	to	to
watch	see	see	watch	see
cartoons	comics	cartoons	animation	cartoons

Table 1: The new query with paraphrase features

To create a paraphrase feature vector for each query, word alignment between a query and its paraphrases is required. We achieve this by having the paraphrase engine output phrase level alignments together with word indices. A simple edit-distance-based approach is then used to align words. The derived alignments may still include one-to-many, many-to-one and many-to-many word mappings. Word frequencies and word order are then applied to create final one-to-one word mapping. If an original word is not mapped to any paraphrase word, "O" is used to indicate the lack of a paraphrase. For example, no word is aligned with "like" in the last column of Table 1.

5. Experiments and results

We evaluate the performance impact of paraphrases on domain, intent and slot models in four different domains: MOVIES, MUSIC, GAMES and APPLICATIONS. Our MOVIES domain dataset contains about 40K queries, GAMES and MUSIC approximately 20K queries each, and the APPLICATIONS domain approximately 5K queries. We do not attempt to handle out-of-domain queries in the present experimental setup.

We first evaluate domain and intent models by comparing our two approaches with baseline models which are trained from original training data using ngram and entity dictionary features. To see the paraphrase impact on different sizes of training data, we ran experiments using fractions of all available data, starting with 10%, 20% through to 100% data. The test set used in all experiments is the same. The overall performance of the domain models is shown in Figure 2.

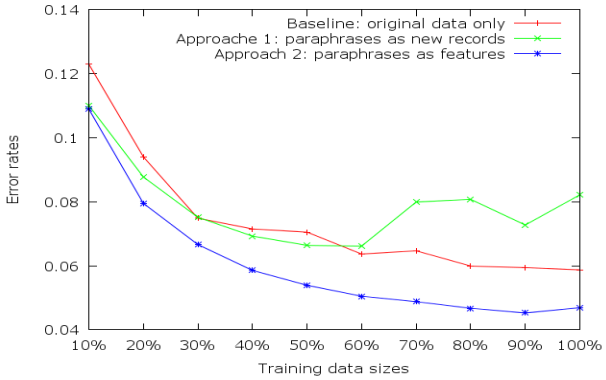

Figure 2: Performance of domain models

Paraphrases are added based on the confidence score assigned by the paraphraser. At most 5 paraphrases per query are added. When a subset of the training data is selected, paraphrases are generated using only that subset; thus "50%" in Figure 2 means only half of training data is used and that paraphrases generated from the same half are added to the augmented training set.

It is evident that the approach that uses paraphrases as features performs best overall. The error rate drops from the baseline consistently. The approach in which paraphrases are added as new records beats the baseline only marginally when less than 60% data is used. Above that level, adding paraphrases to the training set hurts performance. The likely reason is that the paraphrases are not necessary natural, and sometimes are syntactically or semantically incorrect. For example, in paraphrases generated from the sentence"click on firefly trailer", the word "trailer" is replaced with "tow", "truck" and "caravan", none of which are appropriate or relevant.

We also observe that above the 60% threshold, adding more human annotated data to the baseline does not improve the performance. It appears that the model is already stabilizing at that point near its optimal performance over the test set. Thus, adding paraphrases fails to improve coverage and becomes a source of noise in the augmented training sets. Treating paraphrases as features, however, allows noisy terms introduced through paraphrase to be assigned lower weights during the model optimization process.

The overall performance of intent models is shown in Figure 3. Again, we observed similar behavior, where intent models benefit from using paraphrases as features, but adding paraphrases as new records performs below baseline.

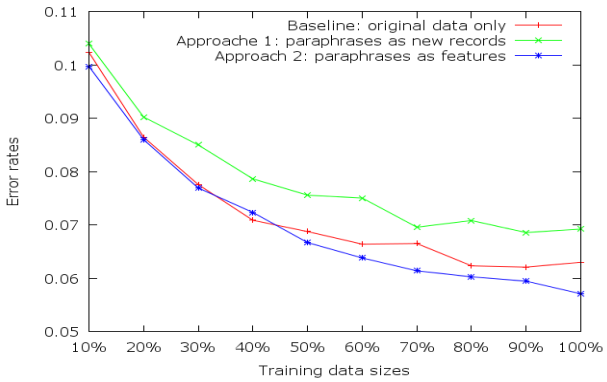

Figure 3: Performance of intent models

In the case of slot models, we used CRF++ [17] to train and test models in the same four domains. We used all training data available and measured the slot model by F1 scores. The results are presented in Table 2. Each test is labeled by training set and feature set. Training sets are labeled as "orig" meaning original queries and "para" meaning generated paraphrases. Feature sets are labeled as "ngram" for ngram feature set, "dic" for dictionary, and "para" for paraphrases. The first experiment "{orig+para} {ngram+dic}" is performed by adding paraphrases to the original training set, using both ngram and dictionary feature sets. The other four rows show results using only original training data, but with different feature sets.

Tests	apps	games	movies	music
{orig+para} {ngram+dic}	85.44	84.90	74.96	65.73
{orig} {ngram}	81.26	87.43	78.32	70.64
{orig} {ngram+dic}	86.24	86.94	83.26	80.15
{orig} {ngram+para}	81.30	88.55	82.92	79.39
{orig} {ngram+dic+para}	**86.90**	**89.25**	**83.43**	**80.45**

Table 2: Performance of slot models

From all slot models in our four domains, we can see that models with paraphrases and ngram features perform better than models with only ngram features. Also models with paraphrases, dictionary and ngram feature sets yield better results than models with only ngram and dictionary features. In other words, adding paraphrase features is always beneficial, no matter whether dictionary features are present.

6. Discussion

The paraphraser used in this paper is generic and has no domain knowledge, with the result that the paraphrases generated are often not appropriate for a given domain. One area for future focus will be to build domain-specific paraphrasers, so that the outputs more closely resemble in-domain queries. This should help reduce noise and generate more (and better) variations at the same time. We also observe that paraphrases of named entities are a major source of noise, owing to limitations in the machine translation alignment algorithm. If the paraphraser has knowledge of application names, movie titles, music titles, game titles, etc., it may be able to either leave the phrases intact or replace them with more robust equivalents.

7. Conclusions

Our experiments demonstrate that using paraphrases as features can improve NLU models for domain and intent detection and semantic slot extraction, and that this method outperforms a simpler approach in which paraphrase is used merely to supplement training data.

8. References

[1] Tur, G. and Mori, R. D., "Spoken Language Understanding: Systems for Extracting Semantic Information from Speech", John Wiley and Sons, 2011.

[2] Vapnik, V,. "The Nature of Statistical Learning Theory", Springer-Verlag, 1995.

[3] Lafferty, J., McCallum, A., and Pereira, F., "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", ICML, 2001.

[4] Quirk C., Brockett, C., and Dolan W. B., "Monolingual Machine Translation for Paraphrase Generation", Association for Computational Linguistics, 2004.

[5] Androutsopoulos I. and Malakasiotis P., "A Survey of Paraphrasing and Textual Entailment Methods", Journal of Artificial Intelligence Research, Volume 38, 135-187, 2010.

[6] Callison-Burch, C., Koehn, P., and Osborne, M. "Paraphrasing with Bilingual Parallel Corpora", HLT/NAACL, 2006

[7] Koehn, P., "Statistical Machine Translation", Cambridge University Press, 2009.

[8] Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. "Statistical Machine Translation for Query Expansion in Answer Retrieval", Proc. of the 45th Annual Meeting of ACL, 2007.

[9] Barzilay, R., McKeown, K. R, "Sentence Fusion for Multidocument News Summarization", Computational Linguistics, 31(3), 2005.

[10] Reiter, E., and Dale, R., " Building Natural Language Generation Systems", Cambridge University Press, 2000.

[11] Kapidakis, S., Mastora, A., and Peponakis, M., "Query Expansion of Zero-hit Subject Searches: Using a Thesaurus in Conjunction with NLP Techniques", Proc. of the second international conference on Theory and Practice of Digital Libraries, 2012.

[12] Hillard, D., Celikyilmaz, A., Hakkani-Tur, D., and Tur, G., "Learning Weighted Entity Lists from Web Click Logs for Spoken Language Understanding", Interspeech, 2011.

[13] Gao, J., He, X., and Nie, J. "Clickthrough-Based Translation Models for Web Search: from Word Models to Phrase Models", CKIM 2010

[14] Gao, J., Xie, S., Xe, X., Ali, A. "Learning Lexicon Models from Search Logs for Query Expansion", EMNLP, 2012

[15] Bannard, C. and Callison-Burch, C., "Paraphrasing with Bilingual Parallel Corpora", Association for Computational Linguistics, 2005.

[16] Koehn, P., Hoang, H., Birch, A., et al., "Moses: Open Source Toolkit for Statistical Machine Translation", Association for Computational Linguistics, 2007

[17] Kudo, T., "CRF++: Yet Another CRF Toolkit", http://crfpp.sourceforge.net, 2009.