Investigating statistical machine translation for query rewrite in Gemini Search

ABSTRACT

Statistical machine translation (SMT) is widely used in human language translation from one language to another one. We found SMT did a good job to find synoynms or near-synoynms for single language. This is a particular merit for query rewrite, especially for sponsored search query rewrite. This work investigates use of SMT and its effects on query rewrite. We got unexpectedly good results.

1. INTRODUCTION

Traditional SMT framework is based on IBM's mathematical translation models. It starts from word alignment. Model parameters for IBM model 1-6 are learned from word alignments. IBM translation model is word based. Modern SMT is based on either phrase or syntax tree, but phrase-based SMT is more popular. While SMT is used for different languages, we found it was useful for the same language to find synonyms and near-synoynms. It is useful for query rewrite, especially for sponsored search because sponsored search has three match type: exact, phrase and broad. SMT-based rewrite is perfect for broad match. SMT rewrite can make query intent unchanged or changed to very minor degree. We will describe the steps to generate query rewrite in the following sections.

2. WORD ALIGNMENT

Word alignment is the first step after query processing such as normalization, punctuation removal, spelling correction, lowercasing, etc. Figure 1 shows three possible alignments given a (query, rewrite) pair, (california, california hotel). alignment (a) is the best option. Word alignment is based on EM algorithm provided that there are a great amout of parallel data. For the above example, the three alignments are equally likely given such a pair. If provied other more pairs such as (california, california motel), (california, california weather), then alignment (a) will be learned as the highest alignment score.

We generate (query, rewrite) pairs from co-bid terms of each ad group. Any two of bid terms are extracted from each ad group and used as paired data in two directions. We used two methods to clean up ad group. (1) if the number of bidterms is more than 50, the ad group are dumped. (2) we calculate cosine similarity of each two of bid terms and aggregated the total sum of cosine similarity and averaged by the total number of pairs. If the value is smaller than a threshold (0.3 used), then dump the ad group. The above two clean up methods only remove 1% paired data. The clean up job is not significant to the final results.

3. PHRASE TRANSLATION TABLE EXTRACTION

Word alignments are made in two directions: from query to rewrite, and from rewrite to query. After word alignments, ngram alignments are extracted and alignment scores are calculated. There are four translation scores are used:

1. f_0: inverse phrase translation probability (f(r|q))
2. f_1: inverse lexical weighting lex(f(r|q))
3. f_2:direct phrase translation probability (f(q|r))
4. f_3: direct lexical weighting lex(f(q|r))

An example is,
california ||| california ||| 0.580602 0.429102 0.578673 0.43034
california ||| california hotel ||| 0.000493097 0.214654 3.87321e-05 0.00190895

In addition to the four translation scores, there are 6 reordering scores that are used to decide phrase position settlement. The 6 features represent:

1. f_4: mono-backward : position not changed with next phrase
2. f_5:swap-backward : position swapped with next phrase
3. f_6:other-backward : position discontiune with next phrase
4. f_7:other-forward : position discontinue with previous phrase
5. f_8:swap-forward : position swapped with previous phrase
6. f_9:mono-forward : position unchanged with previous other phrase

4. DECODING: REWRITE

Given a query, to find the rewrite is to maxize the linear functions:

$$p(r|q) = \sum_i w_i f_i$$

Above, w_i can be learned by optimizing a development set. This work uses default value.

The decoding process is time consuming given huge translation table. Pruning methods are used to avoid greed search.

5. OFFLINE TOP REWRITE SELECTION

Nbest rewrite can be generated from the decoding section. The number of N could reach to 2K (2K reaches memory limit). We have to cut rewrite because (1) most rewrites does not match bid terms. (2) there is a maximum of 50 rewrite for each query according to qbert dictionary.

There are multiple strategies to choose top rewrite. The simplest way is to use top N rewrites based on SMT score. But this method can't diversify ad group. The worst case is all top 50 rewrites are from the same ad group. To maximize ad group coverage and best relevant, we choose rewrite from each ad group equally. We use the following steps:

1. match rewrite with bidterms. Remove non-matched rewrites.
2. Rank rewrites based on query, SMT score (decreasing order)
3. Loop over ad groups. For each loop, output the best matched bidterm. Then remove the best one from next selection. For next loop, output the second best.
4. Loop stopped if 50 rewrites have been output or no new rewrite.

Except the 50 rewrites, we also set a relevance threshold to remove bad rewrites. For each query, we set the highest SMT score rewrite as the base. If a rewrite's score is lower than the base discounted by the threshold, this rewrite is not used. We found using a threshold significantly improved CTR.

6. EXPERIMENTS: SMT REWRITE FOR TRAVEL

To train SMT model for travel queries, we used travel co-bid to generate paired data. We get 169M bidterms. By removing high number bidterm ad group and applied bidterm similarity check, we generated 1.5B bid term pairs. We split these 1.5B bid term pairs into 100 hadoop reducers. Each mapper can handle 15M training pair due to memory limitation. We used online tool Moses to train SMT model in each reducer. Finally we obtained 100 models.

We generated rewrites for 240K travel queries, using 100 models. Each model outputs maximum 2K rewrites (still memory limitation to overpass 2K). In total, 200K rewrites for each query can be obtained by 100 models. By joining with bid terms and top rewrite selection as section 5, we got 7M total rewrites, about 30 rewrites per query. Out of 240K travel queries, there are 5K found no bid terms matched. We will analyze reasons for these unmatched queries.

6.1 CTR evaluation

We used Kunefe May data to calculate metrics. The metrics we used are clicks, impression, CTR, and averaged CPC. CTR equals to total clicks divided by total impressions. CPC is averaged by total sum of single click's CPC devided by all clicks. CTR can evaluate query rewrite relevance. The better relevance, the higher CTR. CPC can evaluate how much Yahoo earns for a single click. We compare SMT rewrites to qbert rewrite using Kunefe Broad match and north first (n1) position. Broad match is Bing match, not Yahoo match. The results are shown in Table 1. SMT rewrite improved CTR and CPC over 10%. qbert has higher impression is a fake number because qbert incorporated Bing's rewrite in this version. Higher CTR proves SMT generated high quality rewrites.

Table 1: online metrics

Rewrite	clicks	impression	CTR	CPC
Qbert	13309	204347	0.065	0.57
SMT	5862	75395	0.077	0.70

Table 2: CTR as number of rewrites

#(rewrite)	10	20	30	40	50
CTR	0.082	0.080	0.079	0.075	0.0749

6.2 SMT score evaluation

We can interpret the SMT score as how likely the rewrite is related to the query. The rewrite with higher score is more relevant to the query than the one with lower score. We can prove this point by CTR. We gradually increased the number of rewrite and calculate CTR in different number. Increasing number of rewrite will include more rewrites with lower SMT score. CTR should be reduced as the number increases. Table 2 shows the results. We did observe the decreasing CTR. In this experiment, we use threshold=-5. Using threshold can remove very bad rewrites. If not, CTR could be much worse for higher number of rewrite.

We used threshold=-5 in top rewrite selection. This threshold is chosen by considering both CTR and impression. If threshold is too high, CTR is better but impression is low.

7. SMT ZERO MATCH ANALYSIS

While hundreds of thousands rewrites are generated for each query, we still found 5K queries out of 24K got no bidterm match. We took some examples and analyzed the following reasons:

1. No demand.
2. Query misspell
3. No deletion model. Query is "2015 broadway shows new york" but bidterm is "broadway shows new york". "2015" is not deleted from rewrite. Another example. Query is "big red bus san francisco". Bidterm is "big bus san francisco" .
4. threshold is too high, especially for single word query. Threshold=-1. Query "airasia" has "airasia flights" matched bidterm. But "airasia flights" score=-1.2 .

Table 3: Choosing threshold

threshold	-1	-2	-3	-4	-5
CTR	0.075	0.083	0.074	0.075	0.075
impression	15579	42351	67774	71791	72410