Learning to Translate: A Query-Specific Combination Approach for Cross-Lingual Information Retrieval

Ferhan Ture
Raytheon BBN Technologies
10 Moulton St
Cambridge, MA, 02138 USA
fture@bbn.com

Elizabeth Boschee
Raytheon BBN Technologies
10 Moulton St
Cambridge, MA, 02138 USA
eboschee@bbn.com

Abstract

When documents and queries are presented in different languages, the common approach is to translate the query into the document language. While there are a variety of query translation approaches, recent research suggests that combining multiple methods into a single "structured query" is the most effective. In this paper, we introduce a novel approach for producing a unique combination recipe for each query, as it has also been shown that the optimal combination weights differ substantially across queries and other task specifics. Our query-specific combination method generates statistically significant improvements over other combination strategies presented in the literature, such as uniform and task-specific weighting. An in-depth empirical analysis presents insights about the effect of data size, domain differences, labeling and tuning on the end performance of our approach.

1 Introduction

Cross-lingual information retrieval (CLIR) is a special case of information retrieval (IR) in which documents and queries are presented in different languages. In order to overcome the language barrier, the most commonly adopted method is to translate queries into the document language. Many methods have been introduced for translating queries for CLIR, ranging from word-by-word dictionary lookups (Xu and Weischedel, 2005; Darwish and Oard, 2003) to sophisticated use of machine translation (MT) systems (Magdy and Jones, 2011; Ma et al., 2012). Previous research has shown that combining evidence from different translation approaches is superior to any single query translation method (Braschler, 2004; Herbert et al., 2011). While there are numerous combination-of-evidence techniques for both mono-lingual and cross-lingual IR, recent work suggests that there is no one-size-fits-all solution. In fact, the optimal *combination weights* (i.e., weights assigned to each piece of evidence in a linear combination) differ greatly across queries, tasks, languages, and other variants (Ture et al., 2012; Berger and Savoy, 2007).

In this paper, we introduce a novel method for *learning optimal combination weights* when building a linear combination of existing query translation approaches. From standard query-document relevance judgments we train a set of classifiers, which produce a unique combination recipe for each query, based on a large set of features extracted from the query and collection. Experimental results show that the effectiveness of our method is significantly higher than state-of-the-art query translation methods and other combination strategies.

2 Related Work

The earliest approaches to query translation for CLIR used machine-readable bilingual dictionaries (Hull and Grefenstette, 1996; Ballesteros and Croft, 1996), achieving around up to 60% of monolingual IR effectiveness. Xu and Weischedel (2005) showed that effectiveness can be increased to around 80% by weighting each translation proportional to its rank in the dictionary. The practice of weighting translation candidates was later formulated as a "structured query", in which each query term is represented by a probability distribution over its translations in the document language (Pirkola, 1998; Kwok, 1999; Darwish and Oard, 2003). Our approach is based on the structured query formulation.

Some of the earliest studies in IR discovered that with different underlying models, the retrieved document set would vary substantially, al-

though the effectiveness was similar (McGill et al., 1979). Later studies showed that combining different representations of the query and/or document often produced superior output (Rajashekar and Croft, 1995; Turtle and Croft, 1990; Fox, 1983). This intuitive idea was supported theoretically by Pearl (1988), concluding that multiple pieces of evidence estimates relevance more accurately, but that the benefit strongly depends on the *quality* and *independence* of each piece. Experiments by Belkin et al. (1995) indicated the need to properly weight each representation with respect to its effectiveness. These so-called "combination-of-evidence" techniques became more powerful with the introduction of `Indri`, a probabilistic retrieval framework specifically designed for combining multiple query and document representations (Metzler and Croft, 2005). Croft (2000) provides a detailed summary of earlier query combination approaches in IR, while Peters et al. (2012) cites more recent related work.

The benefits of combination-of-evidence transfer to the cross-lingual case especially well, since the inherent ambiguity of translation readily provides a diverse set of representations. Most CLIR approaches implement a post-retrieval *merging* of ranked lists, each generated from different query (Hiemstra et al., 2001; Savoy, 2001; Gey et al., 2001; Chen and Gey, 2004) or document (Lopez and Romary, 2009) representations, also called "data fusion". In contrast, we focus on a pre-retrieval combination at the modeling stage, so that a single complex query is used in retrieval, instead of multiple simpler ones. Two advantages of the former are easier implementation (since the approach requires no changes to the modeling side) and the possibly greater diversity that can be achieved by having separate retrieval runs. However, each ranked list needs to be limited in size, which might cause some potentially useful documents not to be considered in the combination at all. Since the focus of this paper is on the modeling end of retrieval, pre-retrieval combination was a more suitable choice, though we think that the two approaches have complementary benefits.

The idea of combining query translations before retrieval has been explored previously. Braschler (2004) combines three translation approaches: output of an MT system, a novel translation approach based on a similarity thesaurus built automatically from a comparable corpus, and a dictionary-based translation. The main reason that this combination does not provide much benefit is due to the lower coverage of the thesaurus-based and dictionary-based translation methods. A similar approach by Herbert et al. (2011) uses Wikipedia to provide translations of certain phrases and entities, and combining that with the `Google Translate` MT system yields statistically significant improvements in English-to-German retrieval. More recently, Ture et al. (2012) presented a more sophisticated translation approach using the internal representation of an MT system, and reported statistically significant improvements when a pre-retrieval combination was performed.

All of the previously cited approaches either use uniform weights for combination, or select weights based on collection-level information. However, as stated previously, numerous studies suggest that certain methods work better on certain queries, collections, languages. In fact, when weights are optimized separately on each collection, they differ substantially across different collections (Ture et al., 2012). For monolingual retrieval, there has been a series of learning-to-rank (LTR) papers that determine weights for query concepts (Bendersky et al., 2011), such that retrieval effectiveness is maximized. A recent study extends this idea to the cross-lingual case, by learning how to weight each *translated word* for English-Persian CLIR (Azarbonyad et al., 2013). In contrast, we extract translated word weights from diverse and sophisticated translation methods, then learn how to weight each *translated structured query*, We call this "learning-to-translate" (LTT), which can be formulated as a simpler learning problem. In CLIR, both LTR and LTT are under-explored problems, with a common goal of applying machine learning techniques to improve query translation, yet with complementary benefits.

To our knowledge, there has been one prior LTT approach: a classifier was trained to predict effectiveness of each query translation, using features based on statistics of the query terms (Berger and Savoy, 2007). Instead of weighting, the translations with highest classifier scores were concatenated, yielding statistically significant improvements over using the single-best translation method. However, the translation methods explored in this paper are all based on one-best MT

systems, making it difficult to draw strong conclusions.

3 Query Translation

The primary contribution of this paper is to show how a diverse set of query translation (QT) methods can be combined effectively into a single weighted structured query, with improved retrieval effectiveness. While our approach can applied to any set of translation methods, we focus on three methods that have complementary strengths and that have shown promise in CLIR: word-based probabilistic translation, one-best MT, and n-best probabilistic MT. We briefly present our implementation of each method; more details can be found in earlier work (Darwish and Oard, 2003; Ture et al., 2012).

Each QT method generates a representation of the query in the document language. In the case of word-based and n-best MT approaches, the representation is a structured query itself, where each query word is represented by a probability distribution over translation alternatives. For one-best MT, the query is represented by a bag of translated words.

3.1 One-Best MT

A query translation approach that has become more popular recently is to simply run the query through an MT system, and use the best output as the query:

$$t_1 t_2 \dots t_l = \mathbf{MT}(s_1 s_2 \dots s_k) \quad (1)$$

where $s = s_1 s_2 \dots s_k$ is the query and $t = t_1 t_2 \dots t_l$ is the translated query.

Since modern statistical MT systems generate high-quality translations for many language pairs, this one-best strategy works reasonably well for retrieval and provides a competitive baseline. A practical advantage of this approach is the ease of implementation – one can simply use any MT interface (e.g., `Google Translate`) as a black box in their CLIR system.

3.2 Probabilistic n-best MT

The top translation might sometimes be incorrect, or might lack some of the alternative representations that are very useful in retrieval. Therefore, considering the n highest scored translations (also referred to as the n-best list in MT literature) has become increasingly popular in CLIR approaches.

In order to benefit from the diversity amongst the n-best translations, one can simply concatenate them together, forming a large list of query terms. However, statistical MT systems also assign probabilities to each translation, which can be incorporated into the query representation for better effectiveness, as suggested by Ture et al. (2012).

In this approach, each of the top n translation candidates from the MT system are processed one by one. For each translation candidate, the MT system provides a translation probability, and alignments between words in the query and its translation. As we process each of the n translations, for each query word s_i, we accumulate probabilities on each translated word t_{ij} aligned to s_i. Finally, we normalize the translation probabilities to get $\mathbf{Pr}_{\text{nbest}}(t_{ij}|s_i)$.

3.3 Word-based

One of the most widely used approaches in CLIR is based on translating each query word s_i independently, with probabilities assigned to each translation candidate t_{ij}. Translations are derived automatically from a bilingual corpus using statistical word alignment techniques, which are used as part of the training of statistical MT systems (Brown et al., 1993). These probabilities can be exploited for retrieval based on the technique of Darwish and Oard (2003) for "projecting" text into the document language. After cleaning up the automatically learned translation probabilities (details omitted for space considerations), we end up with the translation probabilities $\mathbf{Pr}_{\text{word}}(t_{ij}|s_i)$.

4 Combination of Evidence

Once we have multiple ways to represent the query q in the document language ($\text{QT}_i(q), i = 1 \dots m$), it is possible to combine these "pieces of evidence" into a single representation as follows:

$$\text{QT}(q) = \sum_{i=1}^{m} w_i(q)\text{QT}_i(q)$$

and each combination-of-evidence approach differs by how the combination weights w_i are computed:

Uniform In this baseline method, we ignore any information we have about the collection or query and assign equal weights to each method (i.e., $w_i(q) = 1/m$). In our case, this means a weight

of 33.3% to each of the one-best, probabilistic n-best, and word-based QT methods.

Task-specific We can optimize the combination weights by overall effectiveness on a specific retrieval task. Given a query set and collection, we perform a grid search on combination weights (with a step interval of 0.1) and select the weights that maximize retrieval effectiveness. The training is performed in a leave-one-out manner: weights for test query q are optimized on all queries except for q.

Query-specific We propose a novel method to compute combination weights specifically for each query, resulting in a more customized optimization that can take into account how effectiveness of each translation method varies across queries.

In the remainder of this section, we describe the details of our novel query-specific combination method.

4.1 Overview of Query-Specific Combination

We present a novel approach for determining query-specific combination weights by training a classifier for each QT method. Prior to training the classifier, we first run retrieval using each QT method, and evaluate the effectiveness of the retrieved documents. The effectiveness of the i^{th} method on query q (i.e., $f_i(q)$) is then converted into a binary label (further described in Section 4.2). Treating each query as a separate instance, a classifier is trained for each method, generating classifiers $C_1, \ldots, C_m$. During retrieval (i.e., at test time), for each query q, each trained classifier C_i is applied to the query, resulting in a predicted label $l_i(q)$ and the classifier's confidence in a positive label, $C_i(q)$.[1] These values are then used to determine combination weights $w_1(q), \ldots, w_m(q)$ that are custom-fit for the query.

4.2 Labeling

First of all, we discard queries in which the difference between the best and worst performing methods is small (specifically, the worst performing method scores at least $k_1\%$ of the best performing one). For such queries, generating fair training labels is more difficult and therefore more likely to introduce noise into the process.[2] Moreover, these are exactly the queries where choosing optimal combination weights is less important (since all methods perform relatively similarly), so it is reasonable to exclude them from training. In fact, a high number of such queries would indicate lower potential for combination-of-evidence approaches.

For each QT method i, we create training instances per query, per retrieval task. Since our goal is to select the best among existing methods, the training label should reflect the effectiveness of method i relative to other methods. A strategy that we call *best-by-measure* assigns a label of 1 if the effectiveness of the i^{th} method (i.e., $f_i(q)$) is at least $k_2\%$ of the maximum effectiveness for that query, and 0 otherwise. While this directly correlates with retrieval effectiveness, labels might be distributed in an unbalanced manner, which might affect the training process negatively. A balanced labeling requires sorting all training instances by how much better the i^{th} method is than other methods ($\max_{i' \neq i}(f_{i'}(q)/f_i(q))$), and then assigning a label of 1 to the lower half and 0 to the higher half. This strategy is called *best-by-rank*.

4.3 Features

We introduce a diverse set of features, in order to train a robust classifier for predicting when each QT method performs better and worse than others. We split the feature set into four meaningful categories, so that we can measure the impact of each subset separately:

Surface features These features do not require a deep analysis of the query: (a) Number of words in query and the translated query, (b) Type of query that we automatically classify based on predefined templates (e.g., fact question, cause-effect, etc.), and (c) Number of stop words in the query and the translated query.

Parse-based features These features are extracted from a deeper syntactic analysis of the query text: (a) Number of related names found in a named entity database, and (b) Existence of syntactic constituents in query and its translation (e.g., "is there a VVB in the query parse tree").

[1] The confidence in a negative label is $1 - C_i(q)$.

[2] We also experimented with including these queries with a third label (e.g., "same") and train a ternary classifier. Having more labels requires more training data, which is not easy to obtain for this task. Also, obtaining a balanced label distribution becomes even more difficult with three labels.

Translation-based features These features consist of statistics computed from the query and its translation: (a) Number of query words that were unaligned in at least half of the n-best query translations, (b) Number of query words that were aligned to multiple target words in at least half of the n-best query translations, (c) Number of query words that were self-aligned (i.e., target word is exactly same string) in at least half of the n-best query translations, (d) Average / Standard deviation / Maximum / Minimum of entropy of $\mathbf{Pr}_{\text{nbest}}$ of each query word, and (e) Average / Standard deviation / Maximum / Minimum of entropy of $\mathbf{Pr}_{\text{word}}$ of each query word.

Index-based features These features are based on frequency statistics from a representative collection:[3] (a) Average / Standard deviation / Maximum / Minimum of document frequency (df) of query words and their translations, (b) Average / Standard deviation / Maximum / Minimum of term frequency averaged across query words and their translations, and (c) Sum / Maximum / Minimum of total probability assigned to words that do not appear in the collection (df $= 0$).

Additionally, the target language is a default feature in all of our experiments. For each classification task, we train a separate classifier on each subset of these four feature categories, so that there are 16 different sets (including the empty set). After we select which categories to pull features from, we optionally perform feature selection to reduce the number of features by a predefined percentage.

In our experimentation, we observed that collection-based features were most useful for classifying the one-best method, whereas parse-based features were most discriminative for probabilistic 10-best. For the word-based QT method, the translation-based features were most effective in our experiments. We further analyze the effect of various features in Section 5.

4.4 Training and Tuning Classifiers

The `scikit-learn` package was used for the training pipeline (Pedregosa et al., 2012). Using an established toolkit allowed us to experiment with many options for classification, such as the learner type (support vector machine, maximum entropy, decision tree), feature set (16 subsets of the four categories described earlier) and two feature selection methods (recursive elimination or selection based on univariate statistical tests). In the end, we get 96 different parameter combinations while training a classifier for a particular QT method, resulting in the need for tuning — picking the parameters that produce highest accuracy on a representative *tuning set*.

Given that we have a set of queries for testing purposes, there are few strategies for selecting a training and tuning set. One approach is to apply a leave-one-out strategy, so that a classifier is trained and tuned on all but one of the test queries, and then applied on the remaining query to predict its label. We call this the *fully-open* setting.

In a more realistic scenario, there will not be relevance judgments for the test queries, yet there might be a small amount of labeled data similar to the test task (e.g., different queries on same collection) that can be utilized for tuning purposes, and a larger set of training queries from different collections. We call this the *half-blind* setting.

If testing in a new domain, queries of similar type are not available for training and tuning purposes. This is a more challenging scenario than the previous two, yet it is important for real-world applications. In order to demonstrate the effectiveness of the training pipeline in this case, we hold out test queries entirely, then train and tune on queries from a completely different task (i.e., different queries *and* collection). We call this the *fully-blind* setting.

4.5 Retrieval

Once we have classifiers trained for all QT methods, we can apply them to a given query on-the-fly, and compute query-specific combination weights. One approach is *hard weighting*, putting all weight onto a single method — when there are more than one methods classified with label 1, we can either pick one randomly or use the classifier confidence value as a tie-breaker. An alternative is *soft weighting*, where the weight of the i^{th} method can be computed either using classifier confidence C_i (i.e., how confident the model is that the i^{th} method will perform well), precision on tuning set precision_i (i.e., how precise the model is at its pre-

[3] We used the BOLT collection in our experiments.

dictions for the i^{th} method), or both:

$$\begin{aligned} w_i^{s1}(q) =& C_i(q) \\ w_i^{s2}(q) =& \text{precision}_i(1) \times l_i(q) \\ &+(1-\text{precision}_i(0)) \times (1 - l_i(q)) \\ w_i^{s3}(q) =& \text{precision}_i(1) \times C_i(q) \\ &+(1-\text{precision}_i(0)) \times (1 - C_i(q)) \end{aligned}$$

The intuition behind all of these weighting schemes is to produce a weight for each QT method, by taking into account the confidence of the classifier, and/or the precision of the classifier on tuning instances.

The computed weights are normalized before constructing the final query for retrieval:

$$w_i^{\text{final}}(q) = w_i(q) / \textstyle\sum_{j=1}^{m} w_j(q)$$

When compared empirically, we noticed that soft weighting is more effective than hard weighting, as the latter is more sensitive to classifier errors. Among the three soft weighting functions, differences were mostly negligible in our experiments. Hence, we decided to use the simplest weighting function w^{s1}.

4.6 Analytical Model

It is time-consuming to implement various combination-of-evidence approaches and run retrieval experiments. Therefore, it is useful to have an analytical model of the process that can provide a rough estimate of how fruitful it would be to spend this effort, given certain details about the task. The model we present in this section estimates the effectiveness of combining QT methods $1 \ldots m$ on a query set Q, given (1) the effectiveness of each method on Q and (2) error rate of binary classifiers $C_1 \ldots C_m$ on Q. Using this formulation, one can assess the benefit of combination without running retrieval, based only on error rates — this saves precious time during development. Moreover, even without trained classifiers, this model can be used to estimate potential benefits by plugging in *hypothetical* error values. In other words, one can ask the question "If I had classifiers with x% error on this query set, what would be the benefit of using these classifiers to combine QT methods?" before developing any combination approach at all.

The analytical model considers a special case of weighted combination: for each query q, we pick a single QT method $i = 1 \ldots m$, for which the classifier predicts a label of 1. If there are more than one such method, one of them is picked randomly. This simplified version allows us to compute expected effectiveness for q as follows:

$$E[f(q)] = \sum_{\text{method } i} \mathbf{Pr}(\text{pick } i|q) f_i(q)$$

While $f_i(q)$ is an observed value (the effectiveness of the i^{th} method on query q), $\mathbf{Pr}(\text{pick } i|q)$ needs to be estimated (the probability of selecting the i^{th} method). Since this depends on the predicted labels, we consider all possible scenarios $l = l_1 l_2 \ldots l_m$, where each value is the prediction of a classifier. For instance, "l=010" means that classifiers C_1 and C_3 predicted a label of 0, while C_2 predicted a positive label. Marginalizing over the 2^m possible scenarios gives us the following estimate:

$$\begin{aligned} &\mathbf{Pr}(\text{pick } i|q) \\ &= \left(\sum_{l_1=0}^{1} \ldots \sum_{l_m=0}^{1} \mathbf{Pr}(l|q) \right) \times \mathbf{Pr}(\text{pick } i|l, q) \\ &= \left(\sum_{l_1=0}^{1} \ldots \sum_{l_m=0}^{1} \prod_{i=1}^{m} \mathbf{Pr}(l_i|q) \right) \times \mathbf{Pr}(i|l, q) \end{aligned}$$

In the final step, we assumed that classifiers make predictions independent of each other, which is a desired property for successful combination. $\mathbf{Pr}(l_i|q)$ can be estimated using classifier error statistics:

$$\mathbf{Pr}(l_i|q) \sim \frac{\text{count}(\text{predicted} = l_i, \text{true} = l_q)}{\text{count}(\text{true} = l_q)}$$

where l_q is the true label of q. If $l_i = l_q$, this expression becomes the true positive or true negative rate, depending on the value. Similarly, if $l_i \neq l_q$, it is either the false positive or false negative rate.

Finally, the probability that the i^{th} method is selected in a particular scenario depends solely on the predicted labels, since it is a random selection: $\mathbf{Pr}(\text{pick } i|l) = l_i / \sum_{j=1}^{m} l_j$

This concludes the derivation of the analytical model of query evidence combination, which we use in Section 5.1 to evaluate the effectiveness of labeling approaches.

5 Evaluation

We evaluated our approach on four different CLIR tasks: TREC 2002 English-Arabic CLIR, NTCIR-8 English-Chinese Advanced Cross-Lingual Infor-

mation Access (ACLIA), and two forum post retrieval tasks as part of the DARPA Broad Operational Language Technologies (BOLT) program: English-Arabic (BOLT_{ar}) and English-Chinese (BOLT_{ch}). The query language is English in all cases, and we preprocess the queries using BBN's information extraction toolkit SERIF (Ramshaw et al., 2011). State-of-the-art English-Arabic (En-Ar) and English-Chinese (En-Ch) MT systems were trained on parallel corpora released in NIST OpenMT 2012, in addition to parallel forum data collected as part of the BOLT program (10m En-Ar words; 30m En-Ch words). From these data, word alignments were learned with GIZA++ (Och and Ney, 2003), using five iterations of each of IBM Models 1–4 and HMM.

3-gram Chinese and 5-gram Arabic Kneser-Ney language models were trained from the Gigaword corpus (1b words each) and non-English side of the training corpus. Chinese and English parallel text were preprocessed through the Treebank Tokenizer,[4] while no special treatment was performed on Arabic.

For retrieval, we used `Indri`, a state-of-the-art probabilistic relevance model that supports weighted query representations through operators `#combine` and `#weight` (Metzler and Croft, 2005). A character-based index was built for Chinese collections, whereas Arabic text was stemmed using `Lucene` before indexing.[5] English text was preprocessed by `Indri`'s implementation of the Porter stemmer (Porter, 1997). Statistics for each collection and query set are summarized in Table 1.

Before performing any combination, we first ran the three baseline QT methods individually and evaluated the retrieved documents. Mean average precision (MAP) was used to measure retrieval effectiveness, which is a widely used and stable metric, estimating the area under the precision-recall curve. We set $n = 10$ for the n-best probabilistic translation method. Baseline scores are reported in Table 2. The average precision (AP) of each query in these tasks was used to label the query and construct training data accordingly.

In subsequent sections, we evaluate the effect of several variants in the training pipeline.

[4] `http://www.cis.upenn.edu/~treebank`

[5] `http://lucene.apache.org`

5.1 Effect of Labeling

In Section 4.2, we introduced two ways to label instances. In our evaluation, we set the free parameters $k_1 = k_2 = 90$, which filters out 33% of queries from the training set of the BOLT_{ar} task; this percentage is 29% in BOLT_{ch}, 44% in TREC, and 27% in NTCIR.

Labeling determines which query translation method is considered effective or not, which consequently determines what the "learning problem" is (since the objective of the classifier is to separate differently labeled instances). As a result, there are two dimensions to consider when comparing labeling strategies. One is the accuracy of the classifiers on held-out data, and the other is how well the trained classifier reflects this accuracy when used in retrieval. To clarify the distinction, consider a case where every instance is labeled 1. This generates a trivial learning problem with no test errors, yet this does not entail that using these classifiers in retrieval will be more effective than other labeling strategies. If, even with high classifier accuracy, the retrieval effectiveness is low, that indicates a bad choice for labeling.

We can *theoretically analyze* how suitable each labeling method is by applying the analytical model to each CLIR task, setting parameters based on a perfect classifier: true positive/negative rate of 1 and false positive/negative rate of 0 (see Section 4.6). Table 2 shows these results in the "Perfect" column, since these scores represent what *could be* achieved if classifiers were trained to predict labels *perfectly* (no training or retrieval is actually performed). There are two values in each row of the "Perfect" column, one for each labeling strategy. In each row, we found these two values to be statistically significantly higher than any of the baseline scores. This shows that both labeling approaches have the *potential* to improve effectiveness significantly.

We also made an empirical comparison of the two labeling approaches by actually training classifiers with each labeling, and then using the classifiers to combine query translations in retrieval. The "Trained" column in Table 2 shows the MAP we get on each CLIR task (and average classifier accuracies), using either labeling.[6]

Based on these results, we conclude that best-

[6] For a fair comparison, we fixed the train-tune setting to fully-open, trained classifiers on the test collection and reported leave-one-out accuracies.

Lang	Collection		Topics	MT Training data	
	Source	Size (docs)		Source (domain)	Size (words)
Arabic	TREC-02	383,872	50	OpenMT-12 (news/web) BOLT (forum)	10m
Arabic	BOLT	12,258,904	45		
Chinese	NTCIR-8	388,589	100	OpenMT-12 (news/web) BOLT (forum)	30m
Chinese	BOLT	6,693,951	45		

Table 1: Summary of the CLIR tasks in our evaluations.

Task	Baseline			Perfect		Trained	
	one-best	ten-best	word	measure	rank	measure	rank
BOLT_{ar}	0.296	0.311	0.318	*0.341*	*0.341*	0.342 (74)	0.330 (72)
BOLT_{ch}	0.370	0.406	0.407	*0.458*	*0.462*	0.438 (68)	0.426 (60)
TREC	0.292	0.298	0.301	*0.327*	*0.330*	0.305 (59)	0.316 (59)
NTCIR	0.146	0.152	0.141	*0.180*	*0.177*	0.163 (56)	0.162 (61)

Table 2: Retrieval effectiveness of baseline QT methods is presented on the left side, and a comparison of labeling strategies is provided on the right side. All numbers represent MAP values, except for classifier accuracy shown in percentage values (in parantheses). Analytically computed values are shown in italics.

by-measure labeling is more useful in practice, supported by typically higher accuracy and effectiveness. Best-by-rank yields better results only on TREC, but a closer look reveals that the increase in MAP is due to only two outlier queries. For BOLT_{ar}, on the other hand, retrieval with best-by-measure labeling is more effective (statistically significant) than best-by-rank; hence, the former is used in remaining parts of our evaluation.

5.2 Effect of Train-Tune Setting

In Section 4.4, we introduced three major train-tune settings: fully-open, half-blind, and fully-blind. In order to implement these settings, we treat each of the three query sets (BOLT, TREC, NTCIR) as a separate training dataset and experiment with a variety of combinations.

For simplicity, let us demonstrate the variety of experiments assuming the test collection is BOLT. For the fully-open case, the default training data is all of the BOLT queries (this training set is referred to as *b*). Additionally, one can include queries from TREC (referred to as *t*) and NTCIR (referred to as *n*) into the training data. This gives us four different training datasets for the fully-open case: b, $b + n$, $b + t$, $b + t + n$. Similarly, each of the half-blind and fully-blind settings can be applied to three different training sets: For BOLT, these are t, n, $t + n$.[7] This results in ten different experiments run for each task — in each experiment, we train a classifier for each QT method, select the best meta-parameters on the tuning set, and then compute combination weights for retrieval using the classifiers.

Each cell on the left side of Table 3 (under column "Query-specific Combination") shows the results of the most effective experiment for a particular task and train-tune setting. Accuracy values for classifiers varied widely across these experiments. Still, even when accuracies dropped close to or below 50% (i.e. random baseline), combined retrieval was always more effective than any single QT approach, which emphasizes the robustness of our approach. For instance, in the fully-blind setting for the NTCIR task, the individual classifiers had accuracies of only 56%, 49%, and 44% but MAP was 0.163, which is higher than the MAP of any individual method for that collection (0.146, 0.152, or 0.141).

Another key observation in Table 3 is that the domain effect (i.e., training and/or tuning on queries similar to test queries) is only noticeable on the two BOLT tasks. For NTCIR and TREC, we do not observe a boost in MAP when queries from the same task are included in training (i.e., fully-open setting). This can be explained by the BOLT-centric nature of our system components: the text analysis tool and MT systems are tuned mainly for forum data, and the collection-based features are extracted from BOLT. Due to this bias, BOLT queries were most useful in our experiments, supported by the fact that BOLT is always

[7] In the case of half-blind, b is split into two: 20% is used for tuning and the remainder is used for testing.

Task	Query-specific Combination						Uniform	Task-specific	Max
	fully-open		half-blind		fully-blind				
$BOLT_{ar}$	**0.342***†	b	0.330	t+n	0.329	t+n	0.324^{12}	0.329^{1}	0.346
$BOLT_{ch}$	**0.438***†	b	0.428	n	0.426	t+n	0.422^{1}	0.431^{1}	0.466
TREC	0.321	b+t	**0.324***†	b+n	0.321	b+n	0.314^{1}	0.318^{1}	0.332
NTCIR	**0.164***	b+n	0.163	b+t	0.163	b	0.162^{13}	0.162^{13}	0.182

Table 3: A comparison of query combination approaches. For query-specific combination, MAP and training data are shown for the most effective experiment of each train-tune setting. For each task, the highest MAP achieved with our approach is shown in bold. Superscripts 1, 2, and 3 indicate statistically significant improvements over baseline methods one-best, probabilistic 10-best, and word-based, whereas * indicates improvements over all three. Superscript † indicates results significantly better than uniform and task-specific combination methods.

included in the train set when testing on TREC or NTCIR (see lowest two rows in Table 3). Also, when there is no domain effect (i.e., half-blind and fully-blind), more data yields higher effectiveness in 6 out of 8 cases (see two right columns on the left side of Table 3).

5.3 Retrieval Effectiveness

In this section, we compare our novel query-specific combination-of-evidence approach to the baseline CLIR approaches, as well as comparable combination methods (uniform and task-specific combination) in terms of retrieval effectiveness. Based on a randomized significance test (Smucker et al., 2007), the best query-specific combination method (shown in boldface in Table 3) outperforms all baseline QT methods in all tasks with 95% confidence (indicated by superscript * in Table 3). This is not the case for uniform or task-specific query combination, which are statistically indistinguishable from at least one of the QT methods, depending on the task (indicated by superscripts 1, 2, and 3 for one-best, probabilistic 10-best, and word-based QT methods, respectively). When we directly compare our query-specific combination approach to other combination methods, the differences are statistically significant for all tasks but NTCIR (indicated by superscript †).

For reference, we also computed effectiveness for a hypothetical system (denoted by "Max" in Table 3) that could select the best QT method for each query and use only that for retrieval. This is not a strict upper bound, since correctly weighting each method can produce better results, but it is still a reasonable target for effectiveness. In our experiments, Arabic retrieval runs were very close to this target with our combination approach, while the gap for Chinese is still substantial, which is worth further exploration.

6 Conclusions and Future Work

In this paper, we introduced a novel combination-of-evidence approach for CLIR, which *learns a custom combination recipe* for each query. We formulate this as a set of binary classification problems, and show that trained classifiers can be used to produce query-specific combination weights effectively. Our deep exploration of many variants (e.g., labeling, training-tuning, weight computation, analytical formulation) and extensive empirical analysis on four different tasks provide insights for future research on the under-studied problem of combining translations for CLIR.

Our approach advances the state of the art of CLIR, yielding higher effectiveness than three advanced query translation approaches, all based on state-of-the-art MT systems. Furthermore, on three of the four tasks, our combination strategy is statistically significantly better than two comparable combination techniques. Experimental results also suggest that even a uniform combination of query translations is consistently better than any individual method. While it is known that combining translations helps CLIR, we confirm this on a set of modern CLIR tasks, including two target languages and a variety of text domains.

Having a simple linear learning problem allows us to train robust models with relatively simpler features. Nevertheless, we are interested in experimenting with more sophisticated learning approaches. In terms of non-linear classifiers, our experience with decision trees in this paper indicated a higher tendency to overfit. In terms of

combining queries in a non-linear fashion, our future plans include integrating our approach into a LTR framework, and directly optimize MAP. This will also allow us to explore more complex features extracted from query *and* document text, as well as external sources.

Another possible future endeavor is to extend these ideas to (i) other query translation approaches and (ii) document translation. While the exact same problem can be formulated for learning to translate documents effectively, a more complicated infrastructure and longer running times are two challenges that need to be considered.

Finally, we hope this to be a significant step towards more context-dependent and robust CLIR models, by taking advantage of modern translation technologies, as well as machine learning techniques.

Acknowledgments

This work was supported by DARPA/I2O Contract No. HR0011-12-C-0014 under the BOLT program (Approved for Public Release, Distribution Unlimited). The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

Hosein Azarbonyad, Azadeh Shakery, and Heshaam Faili. 2013. Exploiting multiple translation resources for english-persian cross language information retrieval. In *Proceedings of the Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation*, CLEF '13, pages 93–99.

Lisa Ballesteros and W. Bruce Croft. 1996. Dictionary methods for cross-lingual information retrieval. In *Proceedings of the 7th International DEXA Conference on Database and Expert Systems Applications*, pages 791–801.

Nicholas J. Belkin, Paul Kantor, Edward A. Fox, and Joseph A. Shaw. 1995. Combining the evidence of multiple query representations for information retrieval. *Information Processing & Management*, 31(3):431–448, May.

Michael Bendersky, Donald Metzler, and W. Bruce Croft. 2011. Parameterized concept weighting in verbose queries. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 605–614, New York, NY, USA. ACM.

Pierre-Yves Berger and Jacques Savoy. 2007. Selecting automatically the best query translations. In *Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*, RIAO '07, pages 287–300, Paris, France, France. Le Centre de Hautes Etudes Internationales D'Informatique Documentaire.

Martin Braschler. 2004. Combination approaches for multilingual text retrieval. *Information Retrieval*, 7(1-2):183–204, January.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Aitao Chen and Fredric C. Gey. 2004. Multilingual information retrieval using machine translation, relevance feedback and decompounding. *Inf. Retr.*, 7(1-2):149–182, January.

W. Bruce Croft. 2000. Combining approaches to information retrieval. In W. Bruce Croft, editor, *Advances in Information Retrieval*, volume 7 of *The Information Retrieval Series*, pages 1–36. Springer.

Kareem Darwish and Douglas W. Oard. 2003. Probabilistic structured query methods. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 338–344.

Edward A. Fox. 1983. *Extending the Boolean and Vector Space Models of Information Retrieval with P-norm Queries and Multiple Concept Types*. Ph.D. thesis, Cornell University, Ithaca, NY, USA. AAI8328584.

Fredric C. Gey, Hailing Jiang, Vivien Petras, and Aitao Chen. 2001. Cross-language retrieval for the clef collections - comparing multiple methods of retrieval. In *Revised Papers from the Workshop of Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation*, CLEF '00, pages 116–128, London, UK, UK. Springer-Verlag.

Benjamin Herbert, György Szarvas, and Iryna Gurevych. 2011. Combining query translation techniques to improve cross-language information retrieval. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval*, ECIR'11, pages 712–715, Berlin, Heidelberg. Springer-Verlag.

Djoerd Hiemstra, Wessel Kraaij, Renée Pohlmann, and Thijs Westerveld. 2001. Translation resources, merging strategies, and relevance feedback for cross-language information retrieval. In *Revised Papers from the Workshop of Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation*, CLEF '00, pages 102–115, London, UK, UK. Springer-Verlag.

David A. Hull and Gregory Grefenstette. 1996. Querying across languages: a dictionary-based approach to multilingual information retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, pages 49–57.

Kui-Lam Kwok. 1999. English-Chinese cross-language retrieval based on a translation package. In *Workshop on Machine Translation for Cross Language Information Retrieval, Machine Translation Summit VII*, pages 8–13.

Patrice Lopez and Laurent Romary. 2009. Patatras: Retrieval model combination and regression models for prior art search. In *Proceedings of the 10th Cross-language Evaluation Forum Conference on Multilingual Information Access Evaluation: Text Retrieval Experiments*, CLEF'09, pages 430–437, Berlin, Heidelberg. Springer-Verlag.

Yanjun Ma, Jian-Yun Nie, Hua Wu, and Haifeng Wang. 2012. Opening machine translation black box for cross-language information retrieval. In *Information Retrieval Technology*, pages 467–476. Springer.

Walid Magdy and Gareth J. F. Jones. 2011. Should MT systems be used as black boxes in CLIR? In *Proceedings of the 33rd European Conference on Information Retrieval*, ECIR '11, pages 683–686.

Michael McGill, Matthew Koll, and Terry Noreault. 1979. *An Evaluation of Factors Affecting Document Ranking by Information Retrieval Systems*. ERIC reports. School of Information Studies, Syracuse University.

Donald Metzler and W. Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 472–479.

Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2012. Scikit-learn: Machine learning in python. *CoRR*, abs/1201.0490.

Carol Peters, Martin Braschler, and Paul Clough. 2012. *Multilingual Information Retrieval - From Research To Practice*. Springer.

Ari Pirkola. 1998. The effects of query structure and dictionary-setups in dictionary-based cross-language information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 55–63.

M. F. Porter. 1997. Readings in information retrieval. chapter An Algorithm for Suffix Stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

T. B. Rajashekar and W. Bruce Croft. 1995. Combining automatic and manual index representations in probabilistic retrieval. *J. Am. Soc. Inf. Sci.*, 46(4):272–283, May.

Lance Ramshaw, Elizabeth Boschee, Marjorie Freedman, Jessica MacBride, Ralph Weischedel, and Alex Zamanian. 2011. Serif language processing — effective trainable language understanding. In J. Olive et al., editor, *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*, pages 626–631. Springer.

Jacques Savoy. 2001. Report on CLEF-2001 experiments: Effective combined query-translation approach. In *CLEF*, pages 27–43.

Mark D. Smucker, James Allan, and Ben Carterette. 2007. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the 16th ACM conference on Conference on Information and Knowledge Management*, CIKM '07, pages 623–632.

Ferhan Ture, Jimmy Lin, and Douglas W. Oard. 2012. Combining statistical translation techniques for cross-language information retrieval. In *Proceedings of the 24th International Conference on Computational Linguistics*, COLING '12, pages 2685–2702.

Howard Turtle and W. Bruce Croft. 1990. Inference networks for document retrieval. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '90, pages 1–24, New York, NY, USA. ACM.

Jinxi Xu and Ralph Weischedel. 2005. Empirical studies on the impact of lexical resources on CLIR performance. *Information Processing & Management*, 41(3):475–487, May.