Leveraging Data Resources for Cross-Linguistic Information Retrieval Using Statistical Machine Translation

Steve Sloto ssloto@amazon.com
Ann Clifton acclift@amazon.com
Greg Hanneman ghannema@amazon.com
Patrick Porter pwporter@amazon.com
Donna Gates dmg@amazon.com
Almut Silja Hildebrand silja@amazon.com
Anish Kumar anskum@amazon.com

Abstract

Retail websites may provide customers with a localized user experience by allowing them to use a secondary language of preference. Automatic translation of user search queries is a crucial component of this experience. Several domain-adapted SMT systems for search query translation were trained, including language pairs for which smaller-than desired parallel resources were available, such as Polish-German and Chinese-Japanese. We explored several techniques that could be used to optimize MT systems for this use-case. These included specialized forms of pre-processing, such as diacritic normalization and a weak form of language filtering, using byte-pair encoding (BPE) for automatic word segmentation, sampling monolingual query data for use as an LM, and pivoting.

To help measure the impact of these techniques, we also introduced normalized distributed cumulative gain for machine translation (NDCG-MT) as a means to measure the success of our MT system at the downstream information retrieval task. In addition to examining how close our translation is to a human-generated one, we measured the similarity in search results between reference and machine-translated queries.

One additional challenge was the difficulty in choosing a representative sample of user search queries to use as tuning and test data. The most popular search queries may occur significantly more frequently and could include vocabulary likely to be well-covered by the rest of the training data. Consequently, we will also discuss techniques that can be used to optimize selection of tune/test data. In general, we suggest assessing MT performance on both "head queries," those that occur most frequently, and "tail queries," less frequent queries that could be used to evaluate performance on difficult inputs.

1 Introduction

Cross-linguistic information retrieval is an important area for internet content providers in our current multilingual environment. Content providers make information accessible to speakers of other languages through methods such as translating content on individual webpages. Information retrieval systems may also be a part of websites, but these are much more challenging to localize. Translating an entire database, or modifying an IR system for each desired locality would be costly and would be unlikely to scale well. Another approach is to translate the

user queries which are sent to the IR system. High-level descriptions of such systems can be found in many papers, including Nikoulina et al. (2012) who experimentally demonstrated using machine translation to retrieve information on a shared task for cross-lingual search in a library catalog setting, Martin (2016) who experimented with using MT for cross-linguistic information for user support forums, and Guha and Heger (2014) who use machine translation of product searches as part of a wider localization project for a global retail website.

Like numerous other internet content providers, Amazon uses cross-linguistic information retrieval as a component of a secondary language experience that we offer to customers globally. For example, users on marketplace websites such as Amazon.de have the option to browse the retail site in languages such as Czech, Dutch, English, Polish, and Turkish. Search queries from any of these languages can be translated into German and used to query the German product database. This paper focuses specifically on possible ways to optimize statistical machine translation systems for search query translation.

Numerous challenges may be encountered while training these systems. For example, adapting a system for the domain of search query translation may be challenging when significant parallel-data resources are limited. Another potential constraint is a need to optimize MT system performance, such that secondary-language users would experience limited latency. Furthermore, evaluation of these MT systems on their translation quality and performance on down-stream tasks can be important. To this end, an adapted version of the Normalized Discriminative Cumulative Gain Metric may be used to measure MT quality on the downstream information retrieval task. To leverage existing data resources, various techniques were tried, including pivoting, sub-word segmentation (using Byte-Pair Encoding), and different types of pre-processing.

2 Data and Training Setup

2.1 Search Query Data

Search query data has a few unique properties that can make it challenging to use for training MT systems. User search queries are not commonly translated. If one wishes to train an MT system for this domain, one may need to manually sample selections of queries for translation. It may also be desirable to leverage monolingual query data as a system component. Queries are short and can contain numerous untranslatable items and brandnames, which suggest different pre-processing techniques and hyper-parameter settings.

On a randomly chosen day of search traffic, search query data could contain millions of singletons, as well as hundreds of thousands of searches for the most popular queries. Over multiple days of data collection, certain popular queries could dwarf unique user traffic. We will refer to queries that may occur exponentially more frequently as "head queries." We will refer to queries such as these singletons that may occur less often as "tail queries."

If one wishes to use MT as a cross-linguistic IR system component, head queries are paradoxically the most and least important. Knowingly translating the same query hundreds of thousands of time in a day is an inefficient use of computational resources, so popular queries could be handled with a cache instead of being directly translated. Also, vocabulary pertaining to popular queries may be more likely to occur in general purpose training data. In contrast, tail queries may contain typos or fail to retrieve results, but would allow us to examine how a search-query MT system will perform in a "worst case scenario." For the purposes of these experiments, we sought primarily to optimize translations for tail queries, without having a large negative impact on performance on head queries.

For each MT system discussed in this paper, we were able to sample a selection of search query data to be translated by humans and used as tuning and test sets. We sampled tune sets of 2,400 queries, and up to two test sets of 1,000 queries each.

2.2 Other Training Data

In addition to the small amount of human-translated query data, parallel data from a variety of domains was used to train the systems discussed in this paper. Translated catalog data was used, in addition to parallel-data resources drawn from a variety of other domains less related to the query translation task. The distribution of data from these domains varied between language pairs. Systems described in this paper used between 25 and 3 million parallel segments for system training.

2.3 Training SMT Systems

We trained Phrase-Based Statistical Machine Translation systems for search query translation, experimenting with different methods of transforming and filtering the input data. We train various model components for a multi-domain system, such as alignment models, target-side language models, re-ordering models, and translation models.

When an MT system translates a segment, each of these models score possible translations as they are constructed from left to right, using beam search. The models' scores are combined using weights that are automatically learned by tuning to a held-out tune set of human-translated queries. The weighted combination of scores is used to select the best translation.

3 An IR Metric for MT

While conventional automatic evaluation metrics attempt to replicate human judgements of translation quality, these translation systems have a different use case than to have a human read and understand. Consequently, it is advantageous to use an automatic evaluation metric that reflects performance on our downstream task: search. Things that are important to a human about what makes a good translation may differ significantly from what makes a successful search query. Nuances that may be very important to a human in judging quality such as using function words (e.g., making sure to include the article before a noun) or preserving word order (e.g., "phone grey" vs "grey phone") may be irrelevant to returning the most relevant result to a query. Conversely, translation differences that a human may not mind may change the results returned from a query (e.g., "rose" may be a human-admissible synonym for "pink", but may not return the same search results).

The metrics currently used to evaluate MT systems are generally based on measuring the degree of overlap of n-grams between an MT translation and a human-generated reference translation as a way of measuring translation quality. In the case of evaluating MT for search, we want to directly evaluate based on performance on the task we care about, rather than how close our translation is to a human-generated one. We would like to be able to quantify the goodness of an MT translation by comparing the set of results returned by a query translated by our MT system to the results returned by a reference translation query.

In order to do this, for each translation, we make a call to the search index twice: once to retrieve the results for the translated output, and once for the results from the reference translation. We take the top-K results returned for the translated output and evaluate it against the top-K results returned for the reference using a common IR metric, normalized discounted cumulative gain (NDCG), which we have adapted for the MT task.

3.1 NDCG-MT

The basic idea of NDCG is that we want to evaluate the goodness of the results returned from a query based on the relevance score of each query, in proportion to its rank in the list of results returned, since we know that the first result is more important than the tenth, etc.

NDCG was designed to use relevance scores from some gold-standard measure such as human judgments. However, we don't currently have access to this type relevance evaluation

for any arbitrary query/result pairs we with to test. So, we adapt it for our purposes as follows to compute NDCG-MT: instead of human-judgment relevance scores over query results, we take the results returned from the reference translation of a query to be the gold standard. Note that in this approach, we are assuming that whatever results are returned by the reference translation are the gold standard; we do not attempt to measure the actual relevance of the reference query results, or to directly learn a mapping between source-language query inputs and target-language results.

We take the numerical relevance score of a result to be the inverse of the rank of the result in the set of results returned from the reference translation query. So, for example, in a list of ten results returned from the reference query translation, the first result would have a relevance score of 10, and the last would have a score of 1. We can then evaluate the MT system output queries against this scoring. Thus, rather than taking the best possible score as the ideal ordering of the MT output results, we take this from the reference results ranking, so as not to artificially inflate the scoring of the MT results. The full formulation is as follows:

$$\text{NDCG-MT} = \frac{\text{DCG-MT}}{\text{IDCG-MT}},$$

where

$$\text{DCG-MT} = \sum i \in \text{MT-RESULTS} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

and

$$\text{IDCG-MT} = \sum i \in \text{REF-RESULTS} \frac{2^{rel_i} - 1}{\log_2(i+1)},$$

where rel_i denotes the relevance score of result "i".

In addition, we modify the original formulation of NDCG: we base the DCG (discounted cumulative gain) on the MT output results, and the IDCG (ideal cumulative gain) on the reference results ranking, where the original NDCG computes both DCG and IDCG over the same set of results. This is because in the IR tasks for which this metric was originally developed, it was presupposed that there was a fixed set of results, and the DCG and IDCG only differ in how they rank these results. In our case however, we can't assume that the MT output query will return the same set of results as the reference query. In cases where the output translation retrieves no results, it gets a score of 0. When the reference translation doesn't return any results from the index, we disregard the example and it does not contribute to the test set score.

This gives us a tunable metric that can function as a drop-in replacement for BLEU or METEOR, introduced in Papineni et al. (2002) and Banerjee and Lavie (2005) respectively.

4 Pre-Processing Techniques Used To Leverage Data Resources

Since search query translation is vastly different from general purpose machine translation, it may be desirable to preprocess the data to remove differences that might matter to a human, but not to an information retrieval system, such as lowercasing the data and removing diacritics.

By using the NDCG-MT metric, one can directly measure how procedural differences affect the quality of downstream search results.

4.1 Normalization

We tokenized and lowercased the training data for all systems discussed in this paper.

4.1.1 Diacritic Normalization

Many alphabets contain characters with particular diacritic markers added to them to distinguish between sounds. These characters are not always used by speakers of those languages in casual scenarios, such as when typing search queries.

Normalizing diacritic makers for system input could maximize the likelihood of providing a correct translation for these queries. However, it is also possible that words with very different meanings could be normalized to the same form. This potential side-effect could increase ambiguity in the MT system.

If performed, it is optimal to normalize diacritics in a language-specific manner. For example, it is common for German speakers to use 'oe' to represent 'ö'. However, this may not be the case for users with different first languages, such as Turkish or Czech, where 'ö' was more commonly normalized as 'o'. Experimentally normalizing 'ö' to 'oe' for Turkish or other input languages into German, caused a decline in MT performance compared to not normalizing at all.

Table 1 shows the results from experimenting with different normalization heuristics for the Czech-language input for the Czech-German MT system. We built MT systems with the identical hyper-parameter settings and only varied whether diacritic normalization was included as a part of tokenizing system input. The system with the baseline setting did not include any diacritic normalization. The system built with 'strong normalization' normalized all Czech letters with diacritics to the most similar single-letter that did not feature a diacritic. The German-style system did not normalize these characters, and normalized 'ö' to 'oe'. In this case, instead of better leveraging our training data by combining similar representations of words together, we arbitrarily separated these words into distinct forms with no real motivation, with a corresponding performance loss on the test sets.

On our test set of head queries, our normalization-free baseline outperforms all normalization heuristics. For the tail queries, diacritic normalization moderately improves our success on information retrieval and translation, but not to a significant degree. This result suggests that diacritic normalization is most useful for tail-query translation. It also suggests that normalization may be a particularly challenging area for experimentation, since it normalization may substantially hurt general-case performance with marginal worst-case improvement.

Test Set	Normalization	BLEU	NDCG-MT	METEOR	TER	Length
Head	Baseline	55.2	59.6	64.0	36.6	102.9
Head	Strong Normalization	54.0	57.5	62.9	37.7	102.7
Head	German-Style	54.8	59.4	63.1	37.6	103.0
Tail	Baseline	43.3	70.2	62.7	41.4	102.7
Tail	Strong Normalization	43.6	70.9	63.0	41.4	103.0
Tail	German-Style	41.5	70.6	62.0	42.8	103.8

Table 1: Comparison of Different Normalization Heuristics on Test Sets for the Czech-German System

4.2 Language Filtering

Many parallel corpora are noisy, and contain data that is in the incorrect language. For general-purpose MT systems, it may be advisable to filter incorrect language segments out of the training data. User search data can complicate thsi process. As mentioned in Section 2.1, queries are generally short, so a character-based language detection model may have lower accuracy. Search queries also commonly include brand names, which may be in languages other than

those that we are targeting for translation. If "Pomme De Terre" was a French brand name, we may want to include it in non-French training data.

To measure the effect of language filtering on a finished system's MT output quality, Polish-German data was filtered four ways using a character-based language detection library. We tested allowing Polish, German, and English on both sides of the training data, allowing the desired language and English on either side of the training data, allowing only Polish and German on both sides, or allowing only the desired language on either side. Changing the filtering setting had a marginal effect on our overall data size. No more than 1% of the data was discarded compared to our 'weak' baseline. System variants were otherwise identical with respect to hyper-parameter settings, including Byte-Pair encoding. We scored the system against a test set of 1,000 Polish-language queries with corresponding German translations. These queries had been specifically hand-picked to be in the Polish language. Despite the marginal change in training data size, a noticeable effect could be seen in our metrics scores, shown in Table 2.

The more permissive language-detection setting performed best on the downstream information retrieval task, as measured by the NDCG-MT score. Despite the fact that our queries were almost exclusively in the Polish language, retaining a marginally larger variety of untranslatable items in our training data improved our performance.

Allowed Languages	BLEU	NDCG-MT	METEOR	TER	Length	% of Data Used
pl,en,de → de,en,pl	44.1	66.1	57.1	43.6	95.3	100.000%
pl,en → de,en	45.0	65.3	57.0	43.5	95.7	99.929%
pl,de → de,pl	45.2	65.5	56.8	42.9	95.1	99.788%
pl → de	43.7	64.9	56.0	43.5	94.0	99.775%

Table 2: Comparison of Language Filtering Settings on Polish-German System

5 Byte-Pair Encoding

5.1 Motivation & Previous Work

There are many scenarios where we may be interested in examining features below the word level for training MT systems. Some languages, such as Turkish, are morphologically complex, such that a word may contain many affixes. Breaking apart these words into constituent components may enhance translation quality. When translating between related languages, such as Dutch and German, one could be interested in looking at sub-word features to improve performance on transliteration, or other situations in which common changes apply to the input. Lastly, segmentation may also be useful for languages written in characters, such as Mandarin Chinese and Japanese, where discrete spaces between word units may not already exist.

Sennrich et al. (2015) popularized Byte-Pair encoding as an algorithm for unsupervised sub-word segmentation. Intuitively, Byte-Pair Encoding works by breaking apart words into characters, and joining characters into sub-word units based on their most frequently occurring neighbors. This technique does not make use of intuitive phonetic or morphological boundaries, and its output does not look particularly intelligent to human eyes. BPE has been commonly adopted for use in Neural Machine Translations systems, where it reduces vocabulary size and improves performance.

Although Byte-Pair Encoding is commonly used for training NMT systems, it has rarely been used for training statistical models. Kunchukuttan and Bhattacharyya (2016) experimented with using BPE to subsegment data for SMT systems translating between related languages. They built systems for sixteen language pairs in ten different writing systems. They found that BPE works well as a paradigm for segmentation regardless of data size. Byte-Pair encod-

ing out-performed other forms of sub-word segmentation, as well as baseline systems trained without segmentation, and led to an increase in BLEU score in all but one case. They also experimented with different numbers of merge operations for BPE, but tended towards extremely small settings, experimenting between 1k and 4k.

Östling et al. (2017) experimented with using BPE for sub-word segmentation in SMT systems baseline systems trained as part of the University of Helsinki entry into the WMT Shared Translation task in 2017. Their results showed that BPE was not particularly helpful in training SMT models. They attributed this failing to the small number of operations chosen for NMT and not optimized for SMT specifically. Their vocabulary size was around 20,000.

5.2 Our Setup

We used BPE to subsegment our data after it was already tokenized, lower-cased, and pre-processed in any other manner. Our experimental usage of BPE differed from that described by Kunchukuttan and Bhattacharyya (2016) in two ways. First, like Östling et al. (2017), we found that too low of a number of operations led to the possibility of degraded performance. In particular, we found that too low of a setting led to an increased number of garbled translations, in which not all of the BPE-segments in a word were translated. Second, we usually trained BPE using a joint model that learned sub-words from both source and target language training data, instead of using separate models for source and target. Search queries and product names often include non-translatable items that occur in both versions of a parallel segment. Consequently, better performance could be achieved by splitting source and target representations of non-translatable items into the same sub-word units on both halves of our training corpora. For Turkish-German MT, we experimented with training source and target separately, and found that we experienced a loss of 2 BLEU points compared to a joint model. We also experimented with using only a source-side BPE model, and no sub-word segmentation on the target, and experienced a loss of 11.8 BLEU.

After settling on the general experimental setup of training a joint BPE-model as an MT system component, we varied the number of BPE operations. Table 3 shows the result of choosing different numbers of BPE Operations on the quality of MT output measured in BLEU, and Table 4 shows results for the same systems and test sets scored with the NDCG-MT metric. "None" denotes a system without sub-word segmentation, and "0" denotes a system that was built at the character level rather than the word level. For the purposes of this comparison, all systems were built with a re-ordering window of four. All data and hyper-parameter settings are consistent across different versions of a given system.

	0	50,000	100,000	200,00	None
NLNL-DEDE	58.9	63.7	62.5	62.2	61.7
PLPL-DEDE	36.8	43.4	45.1	45.5	42.5
ZHCN-JAJP (Head)	41.5	40.3	40.9	42.2	34.5
ZHCN-JAJP (Tail)	29.6	29.2	28.7	27.5	21.2
CSCZ-DEDE (Head)	49.4	52.8	54.0	52.5	46.1
CSCZ-DEDE (Tail)	29.3	43.0	43.2	43.4	41.7

Table 3: Effect of Different BPE Number of Operations Settings on BLEU Scores

Variants built with BPE strongly outperformed systems that were built without sub-word segmentation. These results suggest that 50,000 or 100,000 Operations may be a suitable baseline setting. 50,000 appears more suitable in cases where both more information is likely to be shared between the source target in segment. It is also worth noting that the NDCG-MT scores

	0	50,000	100,000	200,00	None
NLNL-DEDE	72.2	79.2	78.9	77.3	75.9
PLPL-DEDE	57.8	65.9	67.1	66.2	64.1
ZHCN-JAJP (Head)	57.1	58.3	56.9	56.5	52.3
ZHCN-JAJP (Tail)	45.1	47.5	45.9	45.5	39.9
CSCZ-DEDE (Head)	51.9	57.4	57.5	56.6	55.3
CSCZ-DEDE (Tail)	63.9	70.6	71.3	69.2	66.2

Table 4: Effect of Different BPE Number of Operations Settings on NDCG-MT Scores

shown in Table 4 provide a much cleaner look at the effect of sub-segmenting with different number of BPE operations.

5.3 Interactions Between BPE Model Hyper-Parameter Settings

Utilizing sub-word tokens rather than full-sized word units may affect optimal model hyper-parameter settings, such as the desired size of N-Gram language models, or re-ordering window in re-ordering models. For the specific use-case of search query translation, using BPE did not have as large of an impact on optimal hyper-parameter settings as it may have for other use-cases. For search query translation without BPE, lowering the size of the re-ordering window resulted in roughly equivalent NDCG-MT scores, as well as faster decoding speed. There are two reasons for this: search queries tend to be short, so there are fewer tokens to move around. Also, the order of segments in the MT output is less important in an information retrieval scenario.

Even when breaking up MT system output into smaller sub-word chunks, a relatively small re-ordering window remained an appropriate hyper-parameter choice. Table 5 shows the result of experimenting with the size of the re-ordering window for the Dutch-German system, and resulting systems' performance on BLEU and NDCG-MT. Systems built with BPE were built with 50,000 operations, the setting shown to have the highest performance for Dutch-German in the previous section. The NDCG scores for the system without BPE show best performance with a ROW of 4. For systems built with BPE, there is no real significant difference in IR performance as the ROW is increased, though a ROW of 6 appears to be a good compromise between speed, NDCG-MT, and BLEU.

ROW	BLEU without BPE	NDCG-MT without BPE	BLEU with BPE (50k Ops)	NDCG with BPE (50k Ops)
2	60.1	75.6	62.9	79.1
4	61.7	75.9	63.7	79.2
6	61.7	75.7	64.5	79.6
8	60.8	76.2	63.2	79.7
10	60.9	75.1	63.0	79.9
12	60.9	75.8	63.7	79.3

Table 5: Comparison of Re-Ordering Window Settings With & Without BPE on the Dutch-German System.

6 Using Non-Parallel Data Resources

6.1 Monolingual Data

Among the most common techniques for increasing SMT performance is leveraging monolingual data. For the use-case of search query MT, one can do so by building a language model based on a sample of target-side monolingual search query data. Including this model impacted system performance positively, as shown in Table 6.

System	BLEU	NDGC-MT	METEOR	TER	Length
PLPL-DEDE Including Monolingual Queries	44.1	66.1	57.1	43.6	95.3
PLPL-DEDE Without Monolingual Queries	41.7	64.8	56.3	64.7	95.7

Table 6: Effect of Including Monolingual Search Query Data as a System Component

6.2 Pivoting

Pivoting is an extremely common technique in low-resource scenarios. In general, it has been shown to be useful when there are more substantial amounts of translation data available between the desired source and target languages and a third language, than exist between the source and target languages alone. There are many possible setups for pivoting, including training MT model to translate training data between the pivot language and desired language, or tying together two MT systems in a cascade approach. We experimented with two approaches.

In the first approach, our training corpora were experimentally augmented with data pivoted through English. A simple technique was used, in which we directly pivoted between training data corpora. A desired source-language and target-language string were considered to be a translation if they had an exact string match on an intermediate language string. These segments were grouped with non-pivot segments and added to the training data. All other settings are identical between MT systems including and excluding this pivot data.

We also experimented with cascade approach for Mandarin-Japanese translation. Data was sequentially sent through non-production Mandarin-English and English-Japanese Search Query MT models in sequence. One potentially interesting qualitative side effect of doing this was an increased likelihood of English in the output. The system is forced to translate into English at an intermediate stage. Since there may be numerous Japanese product titles that contain untranslated English, more English appears in the final translation, compared to direct Mandarin-Japanese translation. This result suggests that this technique would be unsuitable for production use without further refinement.

Data	BLEU (Head)	NDCG-MT (Head)	BLEU (Tail)	NDCG-MT (Tail)
Without Pivots	40.5	54.0	28.9	46.5
With Pivots (48% larger)	42.2	56.5	27.5	45.5
Cascading	8.5	20.8	9.7	22.1

Table 7: Comparison of Three Forms of Pivoting on Two Test Sets for Mandarin-Japanese Translation

Table 7 shows a comparison of different pivoting techniques for system performance on Mandarin-Japanese query translation. No system including pivot data outperformed a system without pivot data. There are two main factors that may have contributed to this:

1. Much (though not all) of the pivot data could have come from domains that were farther away from the Search Query Translation task.

2. As was the case for other systems, weak language detection allowing English was used for the Mandarin-English and English-Japanese MT systems. When combining the two in a pivot scenario, the cascade system was considerably more likely to have English in the output. Evidently this setup was extremely harmful for both translation quality, and for downstream search accuracy.

Table 8 shows comparative results over including pivot data in the general PLPL-DEDE training data as scored on an exclusively-Polish test set of 1,000 queries. Including pivot data in the general training data led to a drop in BLEU score, but had no significant impact on the downstream information retrieval task, as measured by NDCG-MT.

Data	BLEU	NDCG-MT	METEOR	TER	Length
Without Pivots	45.2	66.2	57.2	43.2	95.1
With Pivots (15% larger)	44.1	66.1	57.1	43.6	95.3

Table 8: Results of Including Pivot Data in the General Polish-German Training Corpus

7 Conclusions

In this paper, we discussed specific modifications to standard SMT system training in order to optimize for the use-case of search query translation. Several techniques increased the quality of our output in an experimental setting. We summarize these as follows:

- Filtering training data with weak language detection moderately improved system quality compared to stricter filtering. We gained 1.2 NDCG-MT on PL-DE.
- Diacritic normalization potentially improved translation for edge cases, but substantially negatively impacted system performance on "head queries". We gained 0.7 NDCG-MT on a test set of Czech-German tail queries, counterbalanced by a loss of 2.1 NDCG-MT on head queries.
- Using a joint BPE model for sub-word segmentation substantially improved model quality We saw improvements between 1.3 and 6.6 NDCG-MT across language-pairs and test-sets. BPE also outperformed character-based models, though it is possible that those may be competitive for language-pairs such as Mandarin-Japanese and Dutch-German. 50k or 100k operations, and a re-ordering window of 6 is a good baseline for BPE.
- Including monolingual data substantially improved performance. The Polish-German system gained 1.3 NDCG-MT.
- Adding pivot data to the model had mixed results. In the best case, we gained 0.1 NDCG-MT for Polish-German translation, and 2.5 NDCG-MT on Mandarin-Japanese head queries, but lost 1.4 NDCG-MT on Mandarin-Japanese tail queries. A cascade approach to pivoting would be unsuitable without optimizing the constituent MT systems to be used in tandem for this domain.

From the above, it is clear that a combination of sub-word segmentation with BPE and use of monolingual LM data may greatly enhance model performance for this domain. Avoiding strict language filtering may also be useful. Diacritic normalization or the inclusion of pivot data in the general training corpus may be more or less desirable depending on whether one wishes to optimize a system for the translation of head or tail queries.

Lastly, the NDCG-MT metric itself was very useful for evaluating translation quality for the downstream information retrieval task. In particular, NDCG-MT was useful in cases where changes in traditional machine translation metric scores were not reflected by downstream information retrieval quality. For instance, MT metrics alone do not make a compelling case for weak language detection, compared to the NDCG-MT metric, which shows a strong preference for it in table 2. The NDCG-MT score provided much-needed visibility into downstream task performance.

References

Banerjee, S. and Lavie, A. (2005). Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Guha, J. and Heger, C. (2014). Machine translation for global e-commerce on ebay. In *Proceedings of the AMTA*, volume 2, pages 31–37.

Kunchukuttan, A. and Bhattacharyya, P. (2016). Learning variable length units for SMT between related languages via byte pair encoding. *CoRR*, abs/1610.06510.

Martin, R. (2016). Multilingual search with machine translation in the intel communities. In *Proceedings of the AMTA*, volume 2, pages 65–71.

Nikoulina, V., Kovachev, B., Lagos, N., and Monz, C. (2012). Adaptation of statistical machine translation model for cross-lingual information retrieval in a service context. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 109–119. Association for Computational Linguistics.

Östling, R., Scherrer, Y., Tiedemann, J., Tang, G., and Nieminen, T. (2017). The helsinki neural machine translation system. *CoRR*, abs/1708.05942.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.