PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 103 APRIL 2015 85–110

Evaluating Machine Translation Quality Using Short Segments Annotations

Matouš Macháček, Ondřej Bojar

Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics

Abstract

We propose a manual evaluation method for machine translation (MT), in which annotators rank only translations of short segments instead of whole sentences. This results in an easier and more efficient annotation. We have conducted an annotation experiment and evaluated a set of MT systems using this method. The obtained results are very close to the official WMT14 evaluation results. We also use the collected database of annotations to automatically evaluate new, unseen systems and to tune parameters of a statistical machine translation system. The evaluation of unseen systems, however, does not work and we analyze the reasons.

1. Introduction

Manual evaluation is considered as the only source of truth, which automatic metrics try to approximate. However, it suffers from many disadvantages. Since it includes manual labour, it is very costly and slow. Moreover, manual evaluation is not reproducible with exactly the same results; human annotators have different criteria for comparing candidates and even an individual annotator is not consistent with himself or herself in time. Human evaluation is therefore most often used in shared evaluation campaigns and sometimes used by MT developers when a new component of a system needs to be evaluated. It is definitely not feasible to directly use human evaluation in an automatic method for tuning a model's parameters because these methods require to evaluate millions of sentences.

It would be very useful to have an evaluation method with the advantages of both manual and automatic methods. Recently, there actually emerged methods on the

 Corresponding author: bojar@ufal.mff.cuni.cz
Cite as: Matouš Macháček, Ondřej Bojar. Evaluating Machine Translation Quality Using Short Segments Annotations. The Prague Bulletin of Mathematical Linguistics No. 103, 2015, pp. 85–110.
doi: 10.1515/pralin-2015-0005.

boundary of manual and automatic evaluation. See Zaidan and Callison-Burch (2009) and Bojar et al. (2013) for example of such methods. These methods usually require a large manual annotation effort at the beginning to create a database of annotations and then they use the collected database in an automatic way during evaluation of unseen sentences.

The goal of this article is to propose a method, which could be used to manually evaluate a set of systems and the database collected during this manual evaluation could be later reused to automatically evaluate new, unseen systems and to tune the parameters of MT systems. This goal includes, besides proposing the method, also developing an annotation application, conducting a real evaluation experiment and experimenting with reusing the collected database.

The article is organized as follows. In Section 2, we propose the manual evaluation method. In Section 3, we describe our annotation experiment with the proposed method and evaluate annotated systems. We explore the possibility of reusing the collected database to evaluate new systems in Section 4. And finally, we try to employ the collected database in tuning of an MT system in Section 5.

2. SegRanks: Manual Evaluation Based on Short Segments

In the WMT official human evaluation (Bojar et al., 2014),[1] humans annotate whole sentences. They are presented with five candidate translations of a given source sentence and their task is to rank these candidates relative to one another (ties are allowed). One of the disadvantages of this method is that the sentences are quite long and therefore quite hard to remember for the annotators to compare them. Also, when comparing longer sentences, there are many more aspects, in which one sentence can be better or worse than another, and therefore it is more difficult for the annotators to choose the better of the candidates.

To avoid these disadvantages, we propose the following method. Instead of judging whole sentences, we extract short segments[2] from candidates and present them to annotators to rank them. In order to extract meaningful segments with the same meaning from all candidates, we do the following procedure: First, we parse the source sentence and then recursively descend the parsed tree and find nodes that cover source segments of a given maximum length. This is captured in Algorithm 1. Finally, we project these extracted source segments to their counterpart segments in all candidate sentences using an automatic alignment.[3] The whole process is illustrated in Figure 1. This extraction method is inspired by Zaidan and Callison-Burch

[1] `http://www.statmt.org/wmt15` and previous years

[2] The term 'segment' is sometimes used in the literature to refer a sentence. In this article, we will use the term 'segment' for a phrase of few words.

[3] We allow gaps in the projected target segments.

(2009) and by the constituent ranking technique as it was used in WMT07 manual evaluation (Callison-Burch et al., 2007).

Algorithm 1 Short segment extraction from source-side parse tree

```
1: function ExtractSegments(treeNode, minLength, maxLength)
2:     extractedSegments ← list()
3:     leaves ← treeNode.leaves()
4:     if length(leaves) ≤ maxLength then
5:         if lenth(leaves) ≥ minLength then
6:             extractedSegments.append(leaves)
7:     else
8:         for node in treeNode.children() do
9:             segments ← ExtractSegments(child, minLength, maxLength)
10:            extractedSegments.extend(segments)
       return extractedSegments
```

In the constituent ranking in WMT07, these extracted segments were only highlighted and shown to the annotators together with the rest of the sentence. The annotators were asked to rank the highlighted segments in the context of the whole candidate sentences.

We use a different approach here, which is more similar to that used by Zaidan and Callison-Burch (2009). We show the extracted segments without any context and ask the annotators to rank them. The only additional information provided to the annotators is the whole source sentence with the source segment highlighted. The annotators are told that they can imagine any possible translation of the source sentence where the ranked segment fits best. They are instructed to penalize only those segments for which they cannot imagine any appropriate rest of the sentence.

While we are aware that this approach has some disadvantages (which we summarize below), there is one significant advantage: it is much more likely that two systems produce the same translation of a short segment than that they would produce the same translation of a whole sentence. Because we do not show the sentence context, we can merge identical segment candidates into one item, so the annotators have fewer candidate segments to rank. This also allows us to reuse already collected human annotations later to evaluate a new system that was not in the set of annotated systems or to tune parameters of a system.

The following list summarizes known disadvantages of this method. However, we believe that the advantages still outweigh the problems and that our method is worth exploration.

- A system can translate shorter segments quite well but it can fail to combine them properly when creating the whole sentence translation. For instance, a system may fail to preserve the subject-verb agreement. In their paper, Zaidan

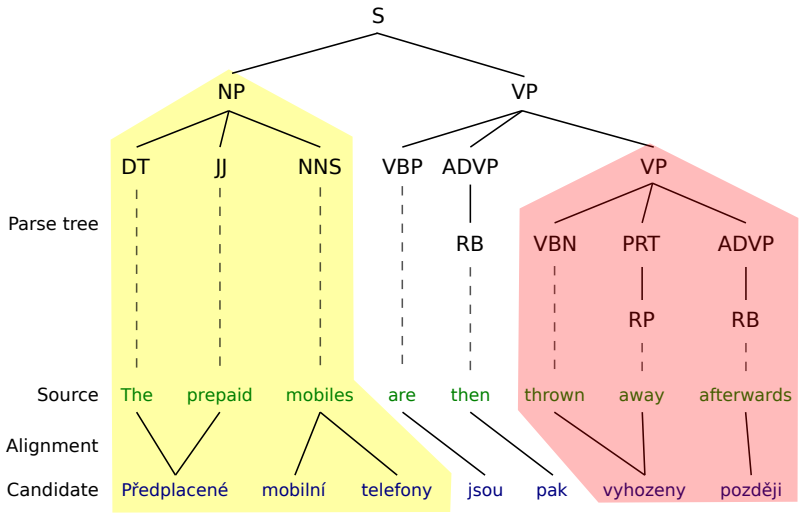

Figure 1: An illustration of candidate segments extraction process. For a given MT system, two segments were extracted from this sentence. The `maxLength` constant was set to the value 4 here, to illustrate that not all of the words are always covered by the extracted segments.

and Callison-Burch (2009) suggest to go up the parse tree and extract also the longer segments that consist of already extracted shorter segments. However, if we use this approach the amount of annotation work would multiply several times. Furthermore, the longer segments are more difficult to rank and the chance that systems' candidates will be identical (so that we can merge them for annotation) is lower.

- The annotators do not see the whole context of annotated short segments. They are instructed to imagine any suitable context of the segment. However, they can fail to imagine a suitable context even if there exists one and wrongly penalize the segment. To partially remedy this disadvantage, we give all extracted short segments to the annotators to judge at once as a source of inspiration.
- Extracted short segments do not cover the whole sentence. For example in Figure 1, the words 'jsou' and 'pak' are not part of any extracted segment. We would avoid this problem if we set the variable `minLength` to zero. This, however, would generate a high number of short segments to annotate.
- Some segments are much more important in conveying the meaning of a sentence than others, and therefore they should not have equal weights when they are considered in scoring. When an annotator ranks system A better than system B in two of three ranked segments, and system B better than system A in

the third segment, it does not always mean that he would have ranked system A better than system B when ranking the whole sentences. The third segment could be much more important for the quality of translation than the first two. We are afraid that it is not possible to easily avoid this problem. However, we also believe that this problem is not so severe and that possible differences in the importance of individual segments cancel out.

- The word-alignments are computed automatically and are not always correct. The projected target segments may not exactly correspond to the source segments and may mislead the annotators.

3. Experiments

3.1. Data and Segment Preparation

We used English to Czech part of the WMT14 (Bojar et al., 2014) test set. We chose this dataset to be able to compare our results with the official WMT14 human evaluation.

The test set consists of 3 003 sentences (68 866 English tokens). It contains both source sentences and reference translations. Besides that, we also used candidate translations of all 10 systems that participated in the English → Czech WMT14 translation task.

The source sentences and all candidate translations were tokenized and normalized. The source sentences were parsed using the Stanford parser. We used lexicalized `englishFactored` model (Klein and Manning, 2003b), which is distributed with the parser. We also tried unlexicalized `englishPCFG` (Klein and Manning, 2003a) and compared the segments extracted using the both parsers on a small random sample of sentences. The `englishFactored` model yielded subjectively better segments.

We constructed the word alignment between the source sentences and the candidate translations using Giza++ (Och and Ney, 2003). Since the alignment algorithm is unsupervised and the amount of all candidate translations is relatively small ($10 \times 3\,003$), we introduced more data by concatenating all candidate translations with 646 605 sentences taken from the Europarl parallel corpus (Koehn, 2005) and with 197 053 sentences taken from the CzEng parallel corpus (Bojar et al., 2012). We used grow-diag-final symetrization method (Koehn et al., 2003) to combine alignments computed in both directions.

We extracted short segments from the parsed source trees using Algorithm 1. The constant `minLength` was set to the value 3 to filter out very short segments, most of which are hard to compare without context. This also helped reduce the number of extracted segments to be annotated. The constant `maxLength` was set to the value 6 so the extracted segments were not too long to compare and at the same time it was more likely that two candidate translations of a segment were equal and thus there would be fewer items to rank (our aim was to make annotations as easy and fast as

possible). We have experimented with various settings of these two constants and the final setting seemed to generate a reasonable number of meaningful segments.

From the 3 003 source sentences, we have extracted 8 485 segments of length 3 to 6 tokens. That is approximately 2.83 segments per sentence on average. The extracted segments covered 54.9 % of source tokens. By projecting the source segments to the candidate sentences using the computed alignments, we obtained $10 \times 8\,485 = 84\,850$ candidate segments. However, after the merging of identical segments, only 50 011 candidate segments were left. This represents 58.9 % of the original candidate segments, or in other words, after the merging we got 5.89 (instead of original 10) candidate segments to be ranked for each source segment on average. These candidate segments were inserted into the database to be ranked by the annotators.

3.2. Ranking of Segments

We have developed a new annotation application called *SegRanks* for this annotation experiment. An example screenshot is given in Figure 2. Annotation instructions were displayed on each annotation screen. This is the English translation of these instructions:

> A number of segments are extracted from the annotated sentence. You are shown a few candidate translations for each of these segments. Your task is to distinguish acceptable candidate translations (the meaning of the segment can be guessed despite a few or more errors) from unacceptable ones (the meaning is definitely not possible to guess from the candidate segment). Also please rank the acceptable candidate translations relatively from the best ones to the worst ones. Please place better candidate translations higher and the worse ones lower. You can place candidates of the same quality on the same rank. We ask that you place unacceptable candidates to the position "Garbage".
>
> Please note that source segments and their candidate translations are chosen automatically and do not have to be perfect. Consider them only as approximate clues. If a candidate segment contains an extra word that does not correspond to the source segment but otherwise could be in the translated sentence, you do not have to rank such candidate any worse. If something is missing in the candidate translation you should consider it an error.

Our goal was to make the annotation as efficient and user friendly as possible. The annotators rank all the segments of a sentence on a single screen (so that they have to read the whole source sentence and reference translation only once). For each annotated segment, they see the source sentence repeated, with the annotated segment highlighted. The annotators rank the segment candidates by dragging and dropping them to appropriate rank positions. When all the candidates of all the source segments of the sentence are ranked, the annotators submit their annotations to the

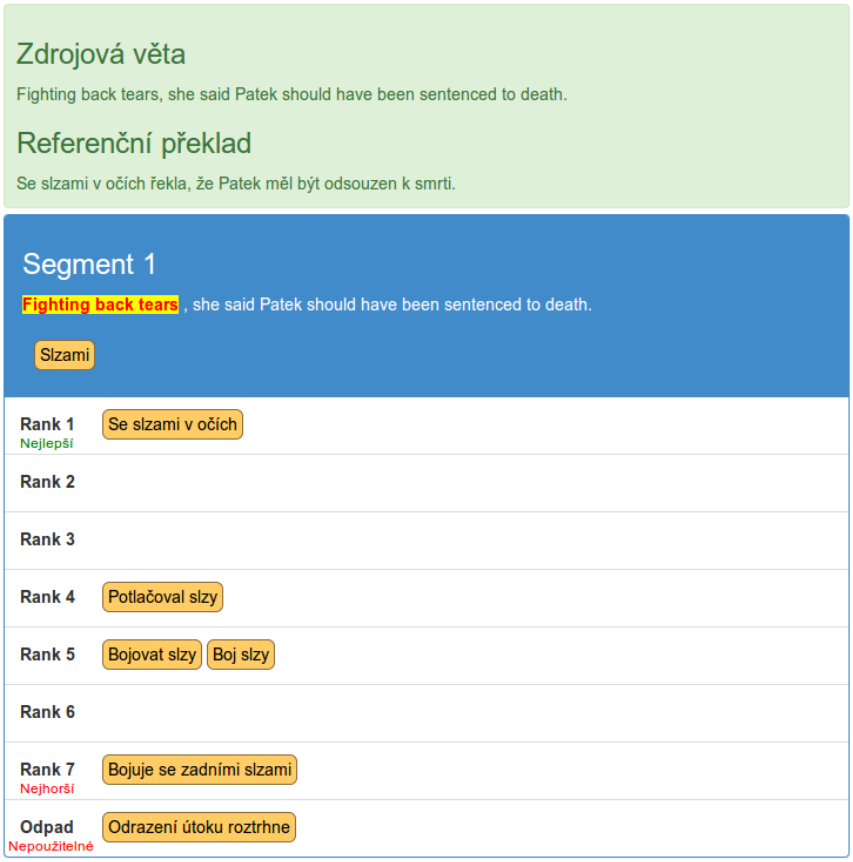

Figure 2: A screenshot of the annotation interface in Czech. The annotators rank the candidate segments by dragging and dropping them into the ranks. The annotators see all annotated segments of a sentence on a single screen.

server. The web interface has a responsive design, so it is displayed correctly on smaller screens, and the drag-and-drop works also on touch screens. The annotators were therefore able to rank segments on a tablet.

During the annotation experiment, 17 annotators ranked segments of 2 765 sentences, which is more than 92 % of the prepared English-Czech test set.

3.3. Annotator Agreements

To measure the reliability and robustness of the proposed annotation method, we compute intra- and inter-annotator agreements. A reasonable degree of these agreements supports the suitability of this method for machine translation evaluation.

We measured the agreements using Cohen's kappa coefficient (κ) (Cohen, 1960). Let $P(A)$ be the proportion of times the annotators agree and $P(E)$ be the proportion of time that they would agree by chance. Then the Cohen's κ is computed using the following formula:

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

In our case, $P(A)$ and $P(E)$ are computed in the context of pairwise comparisons. Approximately 5 % of the annotated sentences were annotated twice by two different annotators (for the inter-annotator agreement). Another 5 % of the sentences were annotated twice by the same annotator (for the intra-annotator agreement). From all the segments of these double annotated sentences, we extracted pairwise comparisons of candidate segments. Then we computed $P(A)$ as the proportion of pairwise comparisons in which annotations match.

We computed the expected agreement by chance as

$$P(E) = P(A > B)^2 + P(A = B)^2 + P(A < B)^2$$

where $P(A = B)$ was computed empirically as the relative frequency of cases where two segments were ranked equaly (across all annotations of all segments of all sentences, regardless the annotator) and the other two were computed as $P(A < B) = P(A > B) = \frac{1-P(A=B)}{2}$. The value of $P(E)$ in our experiment is 0.394, which means that the probability of the outcomes $A > B$, $A = B$ and $A < B$ is not uniform.

The final values of inter-annotator and intra-annotator κ can be found in Table 1. For comparison, we report the corresponding κ values from WMT14 sentence ranking task (Bojar et al., 2014), which were computed identically on the same test set. The exact interpretation of the Kappa coefficient is difficult, but according to Landis and Koch (1977), values in the range 0–0.2 indicate a slight agreement, 0.2–0.4 fair, 0.4–0.6 moderate, 0.6–0.8 substantial and 0.8–1.0 almost perfect agreement. Our method obtains both κ scores better than full sentence ranking in WMT14. However, they are still quite low given that our annotation was designed to be much simpler than the ranking full sentences.

	our method	Bojar et al. (2014)
intra-annotator κ	0.593	0.448
inter-annotator κ	0.397	0.360

Table 1: κ scores measuring intra-annotator and inter-annotator agreements. We also report corresponding κ scores from official WMT translation task for comparison.

3.4. Overall Ranking of Annotated Systems

In the first experiment, we would like to show that the proposed method can be used to produce overall ranking of the annotated systems, which are very similar to the official human evaluation in WMT.

To compute an overall score for each system, we use the method **Ratio of wins (ignoring ties)** from WMT12 (Callison-Burch et al., 2012). We interpret segment ranks as pairwise comparisons: for each short segment candidate we compute how many times it was ranked better than another segment and how many times it was ranked worse. For a given candidate translation C, we can sum up these counts over all short segments of all the sentences to get total number of pairwise wins $win(C)$ and total number of pairwise losses $loss(C)$. The overall score is then computed using this formula:

$$E_{win}(C) = \frac{win(C)}{win(C) + loss(C)}$$

Table 2a reports the overall scores based on the short segment annotations. To compare our method with the classical method of annotating whole sentences, we apply the same method to the annotations collected during WMT14 manual evaluation, see Table 2b.

The overall rankings of the systems obtained from both types of annotation are very similar. However, there are two changes when comparing to the sentence-level results: the system online-B is better and system cu-bojar is worse according to the segments-level results.

3.5. Analysis

For the explanation of the differences between the overall rankings computed on sentence-level and segment-level annotation, we have to look into the data. We extract candidate translations that were ranked high by segment-level annotation but ranked low by the sentence-level annotation. In the following, we present some of these sentences with comments. The ranked segments are in bold.

System	Score
cu-depfix	0.5777
onlineB	0.5642
uedin-unconstrained	0.5626
cu-bojar	0.5606
cu-funky	0.5566
uedin-wmt14	0.5498
onlineA	0.5007
CU-TectoMT	0.4485
commercial1	0.3992
commercial2	0.3492

(a) Short segments annotation

System	Score
cu-depfix	0.6101
cu-bojar	0.6011
uedin-unconstrained	0.5967
cu-funky	0.5823
onlineB	0.5439
uedin-wmt14	0.5285
onlineA	0.5039
CU-TectoMT	0.4473
commercial1	0.3617
commercial2	0.2780

(b) Official WMT14 annotation

Table 2: Overall rankings of systems according to **Ratio of wins (ignoring ties)** score based on (a) our short segment annotations and (b) the official WMT14 full sentence annotations for comparison.

Source: Airlines began charging for **the first and second checked bags** in 2008.
Candidate: Letecké linky začaly nabíjení pro **první a druhý odbavených zavazadel** v roce 2008.
(Sentence 715, online-B)

The translation of the compared segment is relatively good, the case of the noun phrase is wrong but the meaning could be understood. The reason why the whole sentence was ranked poorly is probably the word "nabíjení" ("Charging a battery" in English), which is obviously a wrong lexical choice of the MT system. Unfortunately, this word is not covered by the only ranked segment. A similar problem is also in the following sentence:

Source: I want to fulfil **my role of dad and husband**.
Candidate: Chci, aby splnil **svou roli táty a manžela**.
(Sentence 559, cu-bojar)

The translation of the segment is perfect but the subject of the candidate translation is wrongly expressed as the third person. The whole sentence is therefore correctly ranked as a poor translation. And again, this is not covered by the evaluated segment.

In the two previous examples, the predicate was wrongly translated but unfortunately, it was not covered in the extracted segments and therefore reflected in segment-level annotation. This seems to be the main disadvantage of our method: extracted

segments sometimes do not cover predicates, which are very important for the annotator when judging the overall quality of the sentence.

4. Evaluating New Systems

Machine translation systems often translate a given short source segment identically. This was one of our main motivations for ranking translations of short segments. As we showed in the previous section, evaluating the annotated systems using short segments annotations works reasonably well (despite the shortcomings described above). It would be very useful if we could reuse the database of annotations to evaluate also unseen systems. The more annotated systems we have in the database the more likely it is that an unseen MT system produces a translation of a short segment that is already in the database. We will call this situation a **hit**. If the translated segment is not yet annotated, we will call it a **miss**.

Because we don't have any spare systems that were not annotated, we use a variant of cross validation, called **leave-one-out**: in each step, we choose one system and remove its segments from the database of annotations. Then we treat this system as an unseen one and try to match the system's segments with the segments in the database.

4.1. Exact Matching of Candidate Segments

The most obvious way to evaluate an unseen translation is to compute the **Ratio of wins (ignoring ties)** of all the systems (including the unseen one) only on the hit segments. We extracted pairwise comparisons from all the segment annotations where the segment of the unseen system was a hit and computed the overall score only on such extracted comparisons.

The results of this experiment are given in Table 3. We also report **hit** rate, which is the ratio of hits to all relevant segments (miss + hits).

The average hit rate is 58.8 %, which is above our expectations. However, we see that the hit rate varies considerably across the left-out systems. This is caused by the fact that some systems are very similar; they use the same tools and/or training data. For example all the systems cu-bojar, cu-depfix, cu-funky and both uedin systems are based on the Moses SMT toolkit and their hit rates are very high (0.74–0.93).

As apparent from the table, the obtained orderings of the systems are not very good. The winning system in each of the tables is the one that was left out, which is obviously wrong. Besides that, systems similar to the left-out one get also a much better rank. For example, when the left-out system is one of the systems cu-bojar, cu-depfix and cu-funky, the other two of this group are right below the left-out system. This could be explained by the following statement: MT systems are more likely to agree on a good translation of a segment than on a bad translation. Our technique

system	score
uedin-uncnstr.	0.633
cu-depfix	0.580
uedin-wmt14	0.576
onlineB	0.571
cu-bojar	0.564
cu-funky	0.560
onlineA	0.499
CU-TectoMT	0.425
commercial1	0.377
commercial2	0.329

(a) uedin-uncnstr., hits: 0.67

system	score
commercial1	0.581
uedin-uncnstr.	0.557
onlineB	0.552
uedin-wmt14	0.540
cu-depfix	0.535
cu-funky	0.525
cu-bojar	0.523
onlineA	0.472
CU-TectoMT	0.422
commercial2	0.346

(b) commercial1, hits: 0.28

system	score
commercial2	0.570
onlineB	0.552
uedin-uncnstr.	0.548
cu-depfix	0.535
cu-bojar	0.529
cu-funky	0.524
uedin-wmt14	0.523
onlineA	0.457
CU-TectoMT	0.423
commercial1	0.386

(c) commercial2, hits: 0.28

system	score
CU-TectoMT	0.649
cu-depfix	0.600
cu-bojar	0.584
cu-funky	0.575
onlineB	0.528
uedin-uncnstr.	0.522
uedin-wmt14	0.502
onlineA	0.450
commercial1	0.373
commercial2	0.339

(d) CU-TectoMT, hits: 0.45

system	score
onlineB	0.689
uedin-uncnstr.	0.584
cu-depfix	0.578
cu-funky	0.567
cu-bojar	0.567
uedin-wmt14	0.564
onlineA	0.511
CU-TectoMT	0.406
commercial1	0.360
commercial2	0.320

(e) onlineB, hits: 0.52

system	score
onlineA	0.649
onlineB	0.584
uedin-uncnstr.	0.577
uedin-wmt14	0.568
cu-depfix	0.566
cu-bojar	0.555
cu-funky	0.546
CU-TectoMT	0.411
commercial1	0.365
commercial2	0.318

(f) onlineA, hits: 0.47

system	score
cu-funky	0.630
cu-depfix	0.600
cu-bojar	0.582
uedin-uncnstr.	0.559
onlineB	0.557
uedin-wmt14	0.542
onlineA	0.491
CU-TectoMT	0.446
commercial1	0.378
commercial2	0.331

(g) cu-funky, hits: 0.74

system	score
cu-bojar	0.588
cu-depfix	0.582
cu-funky	0.564
onlineB	0.562
uedin-uncnstr.	0.559
uedin-wmt14	0.545
onlineA	0.498
CU-TectoMT	0.449
commercial1	0.392
commercial2	0.346

(h) cu-bojar, hits: 0.88

system	score
cu-depfix	0.587
cu-bojar	0.570
cu-funky	0.566
onlineB	0.562
uedin-uncnstr.	0.561
uedin-wmt14	0.548
onlineA	0.498
CU-TectoMT	0.449
commercial1	0.394
commercial2	0.345

(i) cu-depfix, hits: 0.93

Table 3: The results of evaluating unseen systems using exact matching and the **leave-one-out** technique. Each subtable is labelled with the left-out system. We also report the hit rates. The table for the system uedin-wmt14 is omitted for the sake of brevity.

thus faces a sparse data issue which affects wrongly translated spans much more than good ones.

To support this statement we have performed an analysis of some sentences from the test set. In the following example sentences, we have marked the hit segments by **bold font** and the missed segments by *italic font*:

Source: *Amongst other things*, it showed that the Americans even monitored **the mobile phone** *of German Chancellor Angela Merkel*.

Candidate: *Kromě jiných věcí* ukázalo, že Američané i sledovali **mobilní telefon** *Germana kancléře Angely Merkelové*.

The missed segment "Kromě jiných věcí" in this sentence is translated quite well (so the above statement does not hold here). However, the only hit segment here "mobilní telefon" is translated correctly and the missed segment "Germana kancléře Angely Merkelové" is not translated correctly. Judging how easy a segment is to translate, is even more difficult than judging the translations. Nevertheless it is obvious that the hit segment "the mobile phone" is easy to translate and the missed segment "of German Chancellor Angela Merkel" is much more difficult to translate. We can see this also in the last example:

Source: They had *searched frantically for their missing dog* and posted appeals **on social networking sites** after she had ran **into the quarry** *following the minor accident*.

Candidate: Měli zoufale *hledal své chybějící psa* a odvolání **na sociálních sítích** poté, co se dostal **do lomu** *po drobné nehody*.

Both of the hit segments are translated very well and we can say that they are also very easy to translate. On the other hand, the missed segments are understandable but not perfect. Compared to the hit segment, they are also more difficult to translate.

Following the manual analysis, we can conclude that MT systems are much more likely to agree on better translations. This however prevents us from matching the segments exactly, because it gives us very overestimated scores for the unseen candidates.

4.2. Matching the Closest Segment by Edit Distance

We would like to compute the scores for all the segments to avoid the problem stated in the previous section. A natural way to approximate the "correct" ranks for unseen segments is to use the rank of the segment from the database with the closest edit distance. We use the character-based Levenshtein distance, which is defined as

the minimum number of insertions, deletions and substitutions of characters required to transform a given string to another:

$$\mathrm{lev}(a, b) = \min_{e \in E(a,b)} \big(D(e) + S(e) + I(e)\big)$$

where $E(a, b)$ denotes a set of all sequences of edit operations that transform the string a to the string b, and $D(e)$, $S(e)$, $I(e)$ denote number of deletions, substitutions and insertions respectively in the sequence s. If more segments in the database have the same minimal distance to the unseen segment, we compute the average of their ranks.

Similarly to the previous experiment, we extracted the pairwise comparisons and computed the **Ratio of wins (ignoring ties)**, see Table 4 for the results. For each left-out system, we also report the average edit distance (AED) of the unseen segments to the closest segments in the database.

The overall rankings of the systems are much more reasonable compared to the exact matching, although they are still not perfect. The scores of systems that were left out are not always the best in the obtained rankings but they are still heavily overestimated. This shows not only that systems are more likely to agree on the better translations than on the worse ones, but also that they produce translations that are closer to better translations than to other translations of similar quality.

The average edit distances vary a lot. The systems cu-bojar, cu-depfix and cu-funky have very low AED (0.2–1.7), because they are very similar to each other. Systems CU-TectoMT, onlineA and onlineB are in the middle of the AED range (3.1–3.9) and since they are quite solitary in the set of ranked systems we can consider their AEDs as representative values. Systems commercial1 and commercial2 have the highest AEDs. This could be explained by the fact that both of the systems are rule based and produce dissimilar translations to those generated by the statistical based systems. It is interesting, however, that their translations are not even similar to each other.

To support the above statement (that the closest segment to an unseen candidate is more likely to be of better quality), we have performed the following analysis: For each left-out system we computed how often the closest segment has better, equal or worse rank than the "unseen" segment. (We can actually do that, because we know the true rank of the segment before it was removed from the database.) We computed these relative frequencies only on the missed segments (the closest segment was not the same segment). The outcomes for the individual systems are given in Table 5. In total, more than a half of the closest segments have a better rank than the original segments and only 20.6 % of the segments have the same rank. This is a very poor result because it means that our approximation method that ranks unseen translations has the accuracy of only 20.6 %. This accuracy does not differ much for individual systems but there is an expected trend of similar systems (cu-bojar, cu-depfix) having this accuracy slightly higher.

Figure 3 plots how these relative frequencies vary with the edit distance. We see that the relative number of closest systems that are ranked better than the original

system	score
uedin-uncnstr.	0.592
cu-depfix	0.576
onlineB	0.562
cu-bojar	0.558
cu-funky	0.555
uedin-wmt14	0.548
onlineA	0.498
CU-TectoMT	0.446
commercial1	0.397
commercial2	0.347

(a) uedin-uncnstr., AED: 2.0

system	score
cu-depfix	0.566
onlineB	0.553
uedin-uncnstr.	0.551
cu-bojar	0.548
cu-funky	0.545
uedin-wmt14	0.537
commercial1	0.495
onlineA	0.488
CU-TectoMT	0.433
commercial2	0.334

(b) commercial1, AED: 5.4

system	score
cu-depfix	0.559
onlineB	0.549
uedin-uncnstr.	0.546
cu-bojar	0.541
cu-funky	0.539
uedin-wmt14	0.533
commercial2	0.484
onlineA	0.483
CU-TectoMT	0.430
commercial1	0.379

(c) commercial2, AED: 5.7

system	score
cu-depfix	0.566
CU-TectoMT	0.557
onlineB	0.554
uedin-uncnstr.	0.552
cu-bojar	0.548
cu-funky	0.545
uedin-wmt14	0.539
onlineA	0.489
commercial1	0.387
commercial2	0.337

(d) CU-TectoMT, AED: 3.9

system	score
onlineB	0.614
cu-depfix	0.574
uedin-uncnstr.	0.559
cu-bojar	0.556
cu-funky	0.552
uedin-wmt14	0.546
onlineA	0.496
CU-TectoMT	0.444
commercial1	0.395
commercial2	0.345

(e) onlineB, AED: 3.1

system	score
onlineA	0.581
cu-depfix	0.570
onlineB	0.556
uedin-uncnstr.	0.555
cu-bojar	0.553
cu-funky	0.549
uedin-wmt14	0.542
CU-TectoMT	0.441
commercial1	0.391
commercial2	0.341

(f) onlineA, AED: 3.4

system	score
cu-funky	0.597
cu-depfix	0.575
onlineB	0.561
uedin-uncnstr.	0.559
cu-bojar	0.557
uedin-wmt14	0.546
onlineA	0.497
CU-TectoMT	0.445
commercial1	0.396
commercial2	0.346

(g) cu-funky, AED: 1.7

system	score
cu-bojar	0.580
cu-depfix	0.576
onlineB	0.562
uedin-uncnstr.	0.561
cu-funky	0.555
uedin-wmt14	0.548
onlineA	0.499
CU-TectoMT	0.447
commercial1	0.398
commercial2	0.348

(h) cu-bojar, AED: 0.4

system	score
cu-depfix	0.579
onlineB	0.564
uedin-uncnstr.	0.563
cu-bojar	0.561
cu-funky	0.556
uedin-wmt14	0.550
onlineA	0.501
CU-TectoMT	0.448
commercial1	0.399
commercial2	0.349

(i) cu-depfix, AED: 0.2

Table 4: The results of evaluating unseen systems using edit distance matching and **leave-one-out** technique. Each subtable is labelled with the left-out system. The abbreviation AED stands for average edit distance, which is computed across all segments. The table for the system uedin-wmt14 is omitted for the sake of brevity.

Unseen system	Worse	Equal	Better
commercial1	28.0 %	18.2 %	53.8 %
commercial2	23.4 %	16.8 %	59.8 %
cu-bojar	22.8 %	30.5 %	46.7 %
cu-depfix	34.2 %	31.4 %	34.4 %
cu-funky	29.3 %	22.9 %	47.8 %
CU-TectoMT	26.2 %	17.8 %	56.0 %
onlineA	28.7 %	19.1 %	52.2 %
onlineB	33.5 %	19.9 %	46.6 %
uedin-unconstrained	32.8 %	21.5 %	45.7 %
uedin-wmt14	32.1 %	21.9 %	46.0 %
All	28.5 %	19.7 %	51.9 %

Table 5: Comparisons of the unseen and the closest segments' ranks. This table shows how often the rank of the closest segment in the database was worse, equal or better than the original rank of the "unseen" segment. These relative frequencies were computed only on the missed segments (which weren't already in the database).

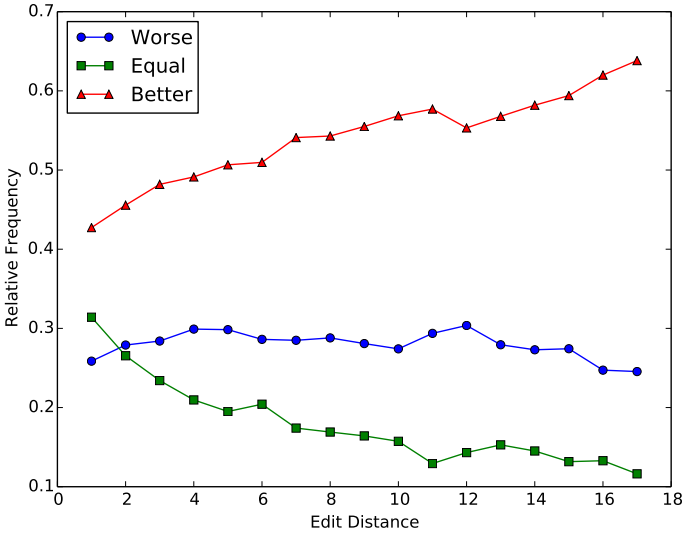

Figure 3: Comparisons of the unseen and the closest segments' ranks with respect to the edit distance. The results in this figure are computed on all the systems.

Unseen segment	Closest segment	D	C
s dokumenty upřednostňujícím doprovodem hradu	se dokumenty favorizovat doprovod hradu	18	**W**
ze 120 domova m2	z 120 m2 domova	7	**B**
vaše ústa ohřívá protein	vaše ústa zahřívání bílkovin	10	**B**
videokonference	videokonferenci	1	**B**
popřel užívání kokainu a	popřela užívání kokainu a	1	**W**
přibližně šedesát- drah kilometru	přibližně šedesát-dráha kilometru	3	**E**
v Liverpoolském porotním soudu	Liverpool Korunního soudu	11	**B**
je nesmysl v gravitaci filmu	Je nesmysl ve filmu gravitace	15	**B**
Pak 64,23 % oprávněných voličů	pak 64,23 % oprávněných voličů	1	**B**
podle DPA agentury	podle DPA agentury té	3	**W**

Table 6: Example candidates and the closest segments. The second last column (**D**) stands for distance and contains distances of the unseen candidates to the closest segments. The last column (**C**) stands for comparison; the closest segment is either worse (**W**), equally good (**E**) or better (**B**) than the original unseen segment.

segment grows quite significantly with the edit distance. The relative number of the worse segments is more or less stable (around 0.3), independent of the edit distance. The relative number of closest segments that are ranked equally as the original segment is decreasing with the edit distance. For example, for the segments whose edit distance to the closest segment is 17, only 10 percent of the closest segments are equally ranked. Relying on similar segments to predict the quality of a new one thus cannot work.

We also list a few example candidate segments in Table 6 together with the corresponding closest segments from the database and their distances. The last column in the table indicates whether the closest segment was ranked better, equal or worse than the "unseen" one.

Unfortunately, we have to conclude that the proposed method, which reuses the database for evaluating unseen translations, does not work. We analyzed the results and the main cause of this failure seems to be that the systems tend to agree on better translations and their translations tend to be more similar to better translations in the database so we cannot predict their rank accurately.

4.3. Enhancing Reference Translations

Following the conclusions from the previous two sections, it seems that errors in machine translation are very unique. Any database of bad examples (bad translations) is therefore very sparse.

In the following experiment, we therefore use only the good candidate segments from the annotated database. The approach used here is, however, different from the previous experiments. We would like to measure how similar candidates are to the good translations from the database. This resembles what automatic metrics do when measuring the similarity between a candidate and reference translations. We have therefore decided to use one of the standard metrics – BLEU – to measure this similarity. First, we are going to introduce the metric and then we are going to customize it for our experiment.

Metric BLEU was developed by Papineni et al. (2002) and it is one of the most popular metrics in the machine translation evaluation. It is defined as the geometric mean of n-gram precisions for $n \in \{1 \ldots N\}$, where N is usually 4. More precisely, for a candidate c and reference translations r_i where $i \in I$, let the clipped count of an n-gram g be defined as follows:

$$\text{count}_{\text{clip}}(g, c, r) = \min\left(\text{count}(g, c),\ \max_{i \in I}(\text{count}(g, r_i))\right)$$

where $\text{count}(g, s)$ denotes the count of n-gram g in the sentence s. The (modified) precision p_n is then defined as:

$$p_n = \frac{\sum_{g \in \text{n-grams}(c)} \text{count}_{\text{clip}}(g, c, r)}{\sum_{g \in \text{n-grams}(c)} \text{count}(g, c)}$$

Using the computed n-gram precisions, we can compute the final BLEU score:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\frac{1}{N}\sum_{i=1}^{N} \log p_n\right)$$

where BP is the brevity penalty (meant to penalize short outputs, to discourage improving precision at the expense of recall) defined as follows:

$$\text{BP} = \begin{cases} 1 & \text{if } |c| > |r| \\ \exp(1 - |r|/|c|) & \text{if } |c| \leq |r| \end{cases}$$

where $|c|$ and $|r|$ are the lengths of the candidate and reference translations respectively. In the case of multiple reference translations, $|r|$ could be the average length, the maximum length or the length closest to the candidate c.

Since the assigned ranks in the database are relative, we cannot know which segments are really good in terms of the absolute quality. We have to assume that there is at least one good candidate translation among the ranked candidates and consider all candidate segments with the best rank as the good translations. We select these candidate segments for each source segment for each sentence.

We use the selected good segments as the reference translations in addition to the original reference sentence translated by a human expert. Since the new references are

System	Score
cu-depfix	0.305
cu-funky	0.302
uedin-unconstrained	0.302
cu-bojar	0.300
uedin-wmt14	0.296
onlineB	0.289
onlineA	0.259
CU-TectoMT	0.225
commercial1	0.176
commercial2	0.160

(a) SegRanksBLEU, correlation: 0.9745

System	Score
cu-depfix	0.221
cu-bojar	0.221
cu-funky	0.221
uedin-unconstrained	0.220
uedin-wmt14	0.215
onlineB	0.207
onlineA	0.187
CU-TectoMT	0.157
commercial1	0.114
commercial2	0.102

(b) BLEU, correlation: 0.9751

Table 7: Overall system ranking according to SegRanksBLEU and BLEU scores. For both of the metrics, we have computed Pearson correlation with human scores.

only short segments and do not cover a whole sentence, we use only the length of the original reference sentence in the computation of the brevity penalty. To distinguish this method from the standard BLEU with the single official reference translation, we will call this method SegRanksBLEU.

Please note, that introducing the new reference translations, which do not change the brevity penalty, can only increase the clipped counts of n-grams occurring in the short segments. Candidates will be rewarded for having n-grams that are also in the good segment translations in the database.

The overall rankings of the evaluated systems as given by SegRanksBLEU and BLEU are in Table 7. As expected, the SegRanksBLEU scores are indeed much higher than BLEU. However, the reported system level correlations of these two metrics are almost equal (correlation of SegRanksBLEU is even a little bit lower). The additional "reference translations" of short segments thus do not bring any useful information.

5. Experiments with MERT

The MERT (Minimum Error Rate Training) method is most often used with BLEU. Recently, there have been a lot of experiments utilizing other automatic metrics. In WMT11 (Callison-Burch et al., 2011), there was a shared task, in which participants tuned a common system with different metrics, and WMT15 plans to run the task again[4]. In WMT11, the tuned systems were then evaluated by humans and some of them outperformed the baseline system tuned with BLEU.

[4] http://www.statmt.org/wmt15/tuning-task/

It is not feasible to employ any sort of human evaluation directly in the MERT process. On one hand, human evaluation is very slow and expensive, on the other hand, MERT requires to evaluate very long n-best list in each iteration. There are some suggestions to do the manual evaluation in a clever way and lower the amount of manual work. For example, Zaidan and Callison-Burch (2009) noticed that a lot of short segments are repeated in a n-best list and therefore suggest to extract these short segments from a n-best list in each MERT iteration and let humans rank them. (Actually our short segment extraction method was partially inspired by this work.) However, they did not try this method in an actual MERT run yet for lack of resources.

We believe that a much less expensive way to introduce an element of human evaluation in the MERT method is to use some sort of semi-automatic metric in which a certain amount of manual work is needed at the beginning and then the metric evaluates translations automatically. In this section, we therefore experiment with previously introduced metrics, which rely on the collected database of short segment ranks.

5.1. Tuned System

The system we tried to tune is the system cu-bojar by Tamchyna et al. (2014), which we also used in previous sections as one of the evaluated systems. This system is Moses-based and combines several different approaches:

- factored phrase-based Moses model
- domain-adapted language model
- deep-syntactic MT system TectoMT

The parallel data used to train the phrase tables consist of 14.83 million parallel sentences taken from the CzEng corpus (Bojar et al., 2012) and 0.65 million sentences taken from the Europarl corpus (Koehn, 2005). The monolingual data used to train language models consist of 215.93 million sentences taken from the Czech side of the CzEng corpus and from 5 sources of a news domain.

We used the implementation of MERT that is distributed with Moses toolkit.[5]

5.2. Metric Variants

Our original idea was that we will use the alignment produced by the Moses decoder when translating the n-best list to project the extracted short source segments to the target side to have candidates that would be extracted the same way as the ranked segments in the database. Unfortunately, the alignment produced by the Moses decoder is very sparse and unreliable (which may be caused by a bug in the code), so we had to get along without the alignment.

The naive approximation is to just test whether the ranked candidate segments from the database also occur in evaluated sentences. The first metric we use in the

[5] http://www.statmt.org/moses/

MERT experiment is therefore very similar to the **Exact Matching** method in Subsection 4.1. For each ranked source segment we test if any of its candidate segments occurs in the evaluated sentence. If it does, we extract all the pairwise comparisons from the matched segment. If more candidate segments occur in the evaluated translation, we choose the longest segment (assuming that the shorter segments are just substrings of the longest one). The final score is then computed as **Ratio of wins (ignoring ties)**. We call this metric ExactMatch in the following.

Even if the hit rate (percentage of segment candidates that are already ranked in the database) computed on a whole n-best list would be 100 %, the ExactMatch metric still does not evaluate whole sentences and could be too coarse and harsh. Moreover, if the hit rate drops during the tuning, the metric score would be computed on a very small percentage of the development set and it would be very unstable. To ensure that the tuning is stable, we interpolate all the metrics in this section (if not said otherwise) together with BLEU with equal weights. We also tried to tune using the ExactMatch metric solely but the tuned system translated very badly and the hit rate dropped very low during the tuning. We could explain this by a hypothesis that the system was tuned to translate a few ranked segments well (so these segments were hits) but other segments were missed and translated badly. The optimal metric score was therefore computed on a very small fraction of the development set and did not reflect the overall quality of the translation.

Since we do not have the alignment and cannot extract the candidate segments exactly, we unfortunately cannot use the method introduced in Subsection 4.2, which matches the closest segment in the database by edit distance. To avoid the shortcomings of the ExactMatch metric (the metric is not computed on all the extracted source segments and an unseen system is more likely to cause a hit in the database with better translations), we propose another variant called PenalizeUnknown. This variant differs from ExactMatch in that it considers all missed segments as the worst translations. We agree that the assumption that all unseen and unranked segments are wrong is not correct, but this approach could increase the hit rate. The question is then whether we prefer to have a system that produces a lot of segments which were already ranked (even badly) or a system that produces a lot of unranked segments, which we hope to be of better quality.

The last variant we experiment with is SegRanksBLEU metric introduced in Subsection 4.3. Because this metric is already based on BLEU metric (and uses the reference translation) we do not interpolate it with BLEU anymore.

5.3. Results and Analysis

We report the results of the tuned system in Table 8. We used the tuned systems to translate the test set (newstest2012) and then we evaluated these translations automatically and manually. For the automatic evaluation we used metrics BLEU (Papineni et al., 2002) and CDER (Leusch et al., 2006). We have also conducted a small scale

Tunable metric	#Mert iterations	Automatic Evaluation BLEU	1-CDER	Manual Evaluation Better	Worse
BLEU (baseline)	11	**0.1782**	**0.3855**	—	—
ExactMatch	20	0.1637	0.3674	22 %	38 %
PenalizeUnknown	8	0.1772	0.3850	**34 %**	**25 %**
SegRanksBLEU	8	0.1753	0.3835	29 %	49 %

Table 8: Results of systems that were optimized to a SegRanks based metric. The items in the first column specify the metric that was used when tuning the system on the development test. The columns BLEU and 1-CDER contain just scores of these metrics computed on the test set translated with the tuned weights. The last two columns contain percentages of better and worse sentences compared to the baseline system in the manual evaluation.

manual evaluation. For each evaluated system, we randomly sampled 100 sentences which were translated differently[6] to the baseline and by the evaluated system. Then we compared manually the sampled sentences with the corresponding translations produced by the baseline system. The task was to choose which sentence is better or whether they are of the same quality. We report how many of the sampled sentences were better and how many of them were worse than the corresponding baseline translation.

None of the metric variants outperformed the baseline system tuned solely to BLEU in the automatic evaluation. This was expected, since the best performing system according to BLEU should be the one that was tuned by BLEU. However, the system tuned by PenalizeUnknown has both the BLEU and CDER scores only a little bit lower.

In the manual evaluation, the only system that outperformed the baseline was the system tuned to PenalizeUnknown. This means that forcing the system to produce known and evaluated segments when translating development set helps to chose better weights. This, however, also means that we discouraged the tuned system to produce unknown and maybe better translations. The best hypothetical translation according to the optimized metric that the tuned system can produce during MERT consists of the best ranked segments from the database. However, there certainly exist better translations.

To see the differences between ExactMatch and PenalizeUnknown, we have plotted the values of hit rates computed in each MERT iteration in Figure 4. PenalizeUnknown gets to the hit rate of 0.7 very quickly (in the sixth iteration) and it's growth is

[6] 58.6 % of all test set sentences were translated differently to the baseline by SegRanksBLEU, 62.4 % by PenalizeUnknown and 83.4 % by ExactMatch

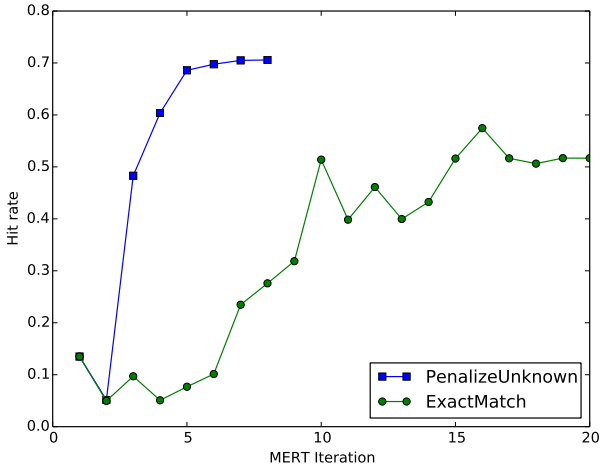

Figure 4: Hit rates computed on the n-best lists produced in each MERT iteration

quite stable. This is expected, since the objective function also indirectly optimizes the hit rate. The hit rate in ExactMatch also grows but much more slowly. It stabilizes around the of value 0.5. It is good that the final value is quite high so the ExactMatch is computed on a non-negligible amount of data. However, it takes many more iterations for ExactMatch to get to the value of 0.5 that it takes for PenalizeUnknown to get to the value of 0.7.

6. Conclusions

In this article, we proposed a new method for manual MT evaluation, called **SegRanks**, in which the annotators rank short segments of a translated sentence relative to each other. Ranking of short segments is easier than ranking of whole sentences and therefore faster. Furthermore, we have developed an easy-to-use and modern annotation interface and conducted a manual evaluation experiment using the proposed method.

When computing overall system scores, the short segment ranks give similar results to the official WMT evaluation with less effort. The measured inter- and intra-annotator κ scores (the normalized agreements) are indeed slightly higher than the corresponding values in the WMT manual evaluation.

To explore the possibility of reusing the collected database of ranked segments to evaluate unseen translations, we have performed several experiments. Although

most of them did not work as expected, we tried to identify and analyze the roots of the failures. The main cause seems to be the fact that each error in machine translation is unique and that segments produced by more than one system are likely to be of better quality. This is also related to another observation we make, that translations are more likely to be closer to better translations than to translations equally good or worse. Maybe, if we had a more dense database (many more than 10 evaluated systems), these phenomena would not influence the results so adversely.

In the last experiment, we tried to use the database to tune a machine translation system using the MERT method. We proposed several variants of **SegRanks**-based metrics adapted for the MERT tuning. The tuned systems were evaluated by humans against the baseline system tuned by BLEU. We were able to improve the tuning of the system using the technique that considered unseen segments as bad and therefore pushed the system to produce known and already evaluated segments.

Bibliography

Bojar, Ondřej, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. The Joy of Parallelism with CzEng 1.0. In *Proceedings of LREC2012*, Istanbul, Turkey, May 2012. ELRA, European Language Resources Association. In print.

Bojar, Ondřej, Matouš Macháček, Aleš Tamchyna, and Daniel Zeman. Scratching the surface of possible translations. In *Text, Speech, and Dialogue*, pages 465–474. Springer, 2013.

Bojar, Ondřej, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, 2014.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) Evaluation of Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W07-0718`.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, and Omar Zaidan. Findings of the 2011 Workshop on Statistical Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July 2011. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W11-2103`.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June 2012. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W12-3102`.

Cohen, J. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37, 1960.

Klein, Dan and Christopher D. Manning. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA, 2003a. Association for Computational Linguistics. doi: 10.3115/1075096.1075150. URL `http://dx.doi.org/10.3115/1075096.1075150`.

Klein, Dan and Christopher D. Manning. Fast Exact Inference with a Factored Model for Natural Language Parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS)*, pages 3–10. MIT Press, 2003b.

Koehn, Philipp. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT, AAMT. URL `http://mt-archive.info/MTS-2005-Koehn.pdf`.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1073445.1073462. URL `http://dx.doi.org/10.3115/1073445.1073462`.

Landis, J Richard and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

Leusch, Gregor, Nicola Ueffing, and Hermann Ney. CDER: Efficient MT Evaluation Using Block Movements. In *In Proceedings of EACL*, pages 241–248, 2006.

Och, Franz Josef and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Comput. Linguist.*, 29(1):19–51, Mar. 2003. ISSN 0891-2017. doi: 10.1162/089120103321337421. URL http://dx.doi.org/10.1162/089120103321337421.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL the 40th annual meeting on Association for Computational Linguistics*, pages 311–318, 2002.

Tamchyna, Aleš, Martin Popel, Rudolf Rosa, and Ondrej Bojar. CUNI in WMT14: Chimera Still Awaits Bellerophon. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 195–200, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W14/W14-3322.

Zaidan, Omar F. and Chris Callison-Burch. Feasibility of Human-in-the-loop Minimum Error Rate Training. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 52–61, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-59-6. URL http://dl.acm.org/citation.cfm?id=1699510.1699518.

Address for correspondence:
Ondřej Bojar
bojar@ufal.mff.cuni.cz
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics,
Charles University in Prague
Malostranské náměstí 25
118 00 Praha 1, Czech Republic