Crowdsourcing High-Quality Parallel Data Extraction from Twitter*

Wang Ling[123] **Luís Marujo**[123] **Chris Dyer**[2] **Alan Black**[2] **Isabel Trancoso** [13]
(1)L^{2}F Spoken Systems Lab, INESC-ID, Lisbon, Portugal
(2)Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA
(3)Instituto Superior Técnico, Lisbon, Portugal
{lingwang,lmarujo,cdyer,awb}@cs.cmu.edu
isabel.trancoso@inesc-id.pt

Abstract

High-quality parallel data is crucial for a range of multilingual applications, from tuning and evaluating machine translation systems to cross-lingual annotation projection. Unfortunately, automatically obtained parallel data (which is available in relative abundance) tends to be quite noisy. To obtain high-quality parallel data, we introduce a crowdsourcing paradigm in which workers with only basic bilingual proficiency identify translations from an automatically extracted corpus of parallel microblog messages. For less than $350, we obtained over 5000 parallel segments in five language pairs. Evaluated against expert annotations, the quality of the crowdsourced corpus is significantly better than existing automatic methods: it obtains an performance comparable to expert annotations when used in MERT tuning of a microblog MT system; and training a parallel sentence classifier with it leads also to improved results. The crowdsourced corpora will be made available in http://www.cs.cmu.edu/~lingwang/microtopia/.

1 Introduction

High-quality parallel data is essential for tuning and evaluating statistical MT systems, and it plays a role in a wide range of multilingual NLP applications, such as word sense disambiguation (Gale et al., 1992; Ng et al., 2003; Specia et al., 2005), paraphrasing (Bannard and Callison-burch, 2005; Ganitkevitch et al., 2012), annotation projection (Das and Petrov, 2011), and other language-specific applications (Schwarck et al., 2010; Liu et al., 2011). While large amounts of parallel data can be easily obtained by mining the web (Resnik and Smith, 2003), comparable corpora (Munteanu and Marcu, 2005), and even social media sites (Ling et al., 2013), automatically extracted parallel tends to be noisy, and, as a result, "evaluation-quality" parallel corpora have generally been produced at considerable expense by targeted translation efforts (Bojar et al., 2013, *inter alia*). Unfortunately, in some domains such as microblogs, the *only* corpora that are available are automatically extracted and noisy.

While phrase-based translation models can effectively learn translation rules from noisy parallel data (Goutte et al., 2012), having a subset of high-quality parallel segments is nevertheless crucial. Firstly, the automatic parallel data extraction system's parameters can be tuned by optimizing on the gold standard data. Secondly, even though the parallel data used to train MT systems can contain a considerable amount of noise, it is conventional to use human annotated parallel data to tune and evaluate the system. Finally, other NLP applications may not be as noise-robust as MT.

We introduce a new crowdsourcing protocol for obtaining high-quality parallel data from noisy, automatically extracted parallel data (§3), focusing on the challenging case of identifying parallel data in microblog messages (Ling et al., 2013). In contrast to previous attempts to use crowdsourcing to obtain parallel data, in which workers performed translation (Ambati and Vogel, 2010; Zaidan and Callison-Burch, 2011; Post et al., 2012; Ambati et al., 2012), our approach only requires that they identify whether a candidate message contains a translation, and if so, what the spans of the translated segments are. This is a much simpler task than translation, and one that can often be completed by workers with only a basic proficiency in the source and target languages.

For evaluation (§4), we use our protocol to build

* A sample of the crowdsourced corpora and the interfaces used are available as supplementary material.

Proceedings of the Ninth Workshop on Statistical Machine Translation, pages 426–436,
Baltimore, Maryland USA, June 26–27, 2014.

parallel datasets on a Chinese-English corpus originally extracted from Sina Weibo and for which we have expert annotations. This lets us quantify the effectiveness of our method under different task variations. We also show that the crowdsourced corpus performs as well as expert annotation (and better than the automatically extracted corpus) for tuning an MT system with MERT. We next apply our method on a corpus of five language pairs (en-ar, en-ja, en-ko, en-ru, en-zh) extracted from Twitter (§5), for which we have no gold-standard data. Using this data in a cross-validation setup, we train and evaluate a maxent classifier for detecting parallel data (§6), and then we conclude (§7).

2 Related Work

Our work crosses crowdsourcing techniques and automatic parallel data extraction from microblogs. In this section, we shall provide background information and analysis of the work performed in these two fields.

2.1 Parallel Data Extraction from Microblogs

Many sources of parallel data exist on the web. The most popular choice are parallel web pages (Resnik and Smith, 2003), while other work have looked at specific domains with large amounts of data, such as Wikipedia (Smith et al., 2010). Microblogs, such as Twitter and Sina Weibo, represent a subdomain of the Web. Some of its characteristics is the informal language used and the short nature of the messages that are posted. Due to its large size and growing popularity, work has been done on parallel data extraction from this domain. Ling et al. (2013) attempt to find naturally occurring parallel data from Sina Weibo and Twitter. Some examples of what is found are illustrated in Figure 1. The extraction process starts by finding the parallel segments within the same message and the word alignments between those segments that maximize a hand-tuned model score.

Another method (Jehl et al., 2012) leverages CLIR (Cross Lingual Information Retrieval) techniques to find pairs of tweets that are translations. The main challenge in this approach is the large amount of pairs of tweets that must be considered, which raises some scalability issues when processing billions of tweets.

Our crowdsourcing method can be applied to annotate data from any naturally occurring source. In this paper, we will use the corpus developed by Ling et al. (2013), since it is publicly available and has parallel data for 6 languages from Twitter, and for 10 languages from Sina Weibo.

2.2 Parallel Data using Crowdsourcing

Most of the work done in building parallel data using crowdsourcing (Ambati and Vogel, 2010; Zaidan and Callison-Burch, 2011; Post et al., 2012; Ambati et al., 2012) relies on using crowdsourcing workers to translate. These methods must address the fact that workers may produce poor and sometimes incorrect translations. Thus, in order to find good translations, subsequent postediting and/or ranking is generally necessary.

In contrast, in our work, crowdsourcing is used for data extraction rather than translation, a substantially simpler task than translation (in particular, translation of informal text) that requires less expertise in the language pair (basic proficiency in the two languages is generally sufficient to successfully complete the task). Furthermore, assessing whether a worker performed the task correctly and combining the outputs of different workers is simpler. The time spent per item is also reduced: our annotation interface only requires the worker to make a few clicks on the tweet to complete each annotation, meaning that tasks are completed faster and with less effort, allowing us to obtain translations at lower cost. On the other hand, the main drawback of our method is that it can only obtain parallel data from translations that exist, which corresponds to the amount of posts that have been translated and posted. This limits the potential coverage of our method. Furthermore, the resulting datasets may not be fully representative of the Twitter domain, since not all types of content are translated and follow the same distribution as the data in Twitter.

3 Proposed Crowdsourcing Protocol

As discussed above, automatically extracted parallel is often noisy. The sources of error range from language detection errors, to errors determining if material is actually translation, and errors in extracting the appropriate spans of the translated material. Consider the fragment of the microblog parallel corpus mined by Ling et al. (2013), which is shown in Figure 1. In the Korean-English message, the system may incorrectly added the untranslated word *Hahah* in the English segment,

and missed the translated word *Weather*. At a high level, the task faced by annotators will be to identify and resolve such errors.

3.1 Overview

We separate the tasks of identifying the parallel posts, which we shall denote by **identification**, and of locating the parallel segments, which we will call **location**. The justification for this is that the majority of the tweets are not parallel, as reported by Ling et al. (2013), and the location of the parallel data is only applicable if the tweet actually contains parallel data. This is also desirable because the identification task is simpler than the location task. Firstly, identifying whether a tweet contains translations requires much less proficiency in the respective languages than locating the parallel segments, since it only requires the worker to understand parts of the message. This means we can have more potential workers capable of performing this task. Secondly, the first task is a binary decision, and each annotation can be completed with only one action, which means that the average required time for this task is much lower than the second task and the payment required for each hit will naturally be lower as well. Finally, combining worker results for a binary decision is simpler than combining translations, since the space of possible answers is several orders of magnitude lower.

As crowdsourcing platform, we use Amazon's Mechanical Turk. In this platform, the requesters can submit **tasks**, where one can define the number of workers n that will complete each task and what is the payment p for each task submission, henceforth denoted as **job**. In our work, we had to consider the following components:

- **Interface** - To submit a task, an interface must be provided, which workers will be using to complete the job.
- **Worker Quality Prediction** - After submitting a job, the requester can accept and pay the agreed fee or reject the task. It is crucial to have a method to automatically predict whether workers have performed the job properly, and reject them otherwise.
- **Result Combination** - It is common for multiple workers to complete the same task with different results. Thus, a method must be implemented to combine multiple responses for correctly predicting the desired response.

We structured each of our tasks as a series of q questions, which include a small number of references r, for which we know the answers. Thus, the amount of answers we obtain for each dollar is given by $\frac{q-r}{np}$, where n is the number of workers per task and p is the payment for each task. In order to maximize this quotient, we can either reduce the number of reference question r, the number of workers per task n, or the payment p. However, reducing r will also limit our capability of estimating the quality of the worker results, since we will have less data to make such prediction. For the same reason, reducing n will limit our ability to combine results properly. As for the payment p, while there is no direct effect on our task, it has been noted that workers will perform the task faster for higher payments (Post et al., 2012). In our work, we will propose methods to predict quality and combine results that will minimize the requirements for n and r, while maximizing the quality of the final results.

3.2 Parallel Post Identification

In the identification task, for each question, we will show a post, and solicit the worker to detect if it contains translations in a given language pair.

Interface The interface for this task is straightforward. We present to the worker each tweet individually, together with a checkbox to be checked in case the tweet contains parallel data. The navigation between tweets is done by adding next and previous buttons, allowing the user to go back and review previous answers. Finally, the worker can only submit the HIT after traversing all 25 questions. Unlike the work in crowdsourcing translation (Zaidan and Callison-Burch, 2011), where automatic translation systems are discouraged, since it produces poor output, we allow its usage as long as this leads to correct annotations. In fact, we add a button to automatically translate the tweet into English from the non-English language.

Worker Quality Prediction We accept the job if it answers enough reference questions correctly. We consider two different approaches to select references. A random sampler that selects tweets randomly and a balanced sampler that selects the same number of positive and negative samples. As notation, we will denote as acceptor

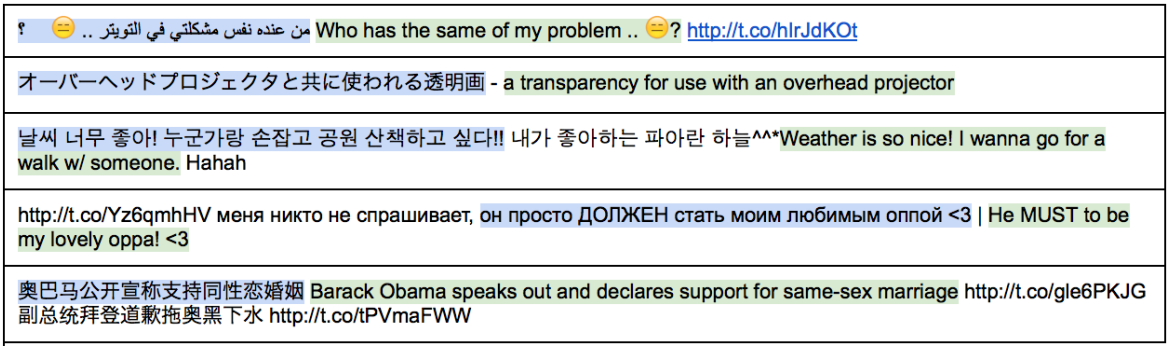

Figure 1: Parallel microblog posts in 5 language pairs. Shaded backgrounds mark the parallel segments (annotated manually), non shaded parts do not have translations.

$accept(rand, c, r)$ a setup where the worker's job is accepted if c out of r randomly sampled references are correctly answered. Likewise, acceptor $accept(bal, c, r)$ denotes the same setup using balanced reference questions.

Result Combination Given n jobs with answers for a question that can be either positive or negative, we calculate the weighted ratio of positive answers, given by $\frac{\sum_{i=1..n} \delta_p(i)w(i)}{\sum_{i=1..n} w(i)}$, where δ_p is one if answer i is positive and 0 otherwise, and $w(i)$ is the weight of the worker. $w(i)$ is defined as the ratio of correct answers from job i in the reference set. If the weighted ratio is higher than 0.5, we label the tweet as positive and otherwise as negative.

3.3 Parallel Data Location

In the location task, we also present one tweet per question, where the worker will be asked to identify the parallel segments. The worker can also define that there are no translations in the tweet.

Interface The interface for this task presents the user with one tweet at a time, and allows the user to break the tweet into segments, by clicking between characters. Each segment can then be classified as English, the non-English language (Ex: Mandarin), or non-parallel, which is the default option. To understand the concept of non-parallel segments, notice that when we are locating parallel data in tweets, we are essentially breaking the tweet into the structure "N_{left} P_{left} N_{middle} P_{right} N_{right}", where P_{left} and P_{right} are the parallel segments and N_{left}, N_{middle} and N_{right} are textual segments that are non-parallel. These may not exist, for instance, the Arabic tweet in Figure 1 (line 1) does not contain any non-parallel text and does not require any non-parallel segments to delineate the parallel data. The Korean tweet (line 2), on the other hand, has an N_{middle} corresponding to *내가좋아하는파아란하늘^^** and an N_{right} corresponding to *Hahah* and requires two non-parallel segments to locate the parallel data.

Thus, if the worker does not commit any errors, each question can be answered with at most four clicks, when all five segments exist, and two option choices for identifying the parallel segments. In the easiest case, when only the parallel segments exist, only one click and two option choices are needed. If there are no translations, the button *no translations* can be clicked.

For instance, to annotate the Korean tweet in Figure 1, the worker must click immediately before *내가*, then before *Weather* and finally before *Hahah*. Then on the drop-down box of the first and and third segments, the worker must choose Korean and English, respectively. The interface after these operations is show in Figure 2.

Work Quality Prediction To score the worker's jobs, we use the scoring function devised in (Ling et al., 2013), which measures the word overlap between the reference parallel segments segments and the predicted segments. However, setting the score threshold to accept a job is a challenge, since scores are bound to change for different language-pairs and domains. Moreover, some tweets are harder to annotate than others. Learning this threshold automatically requires annotated data, which we do not have for all language pairs and domains. Thus, we propose a method to generate thresholds specifically for each sample.

We consider a "smart but lazy" pseudo worker, who will complete the same jobs automatically and generate scores that the real worker's jobs must beat to be accepted. We say he is "smart",

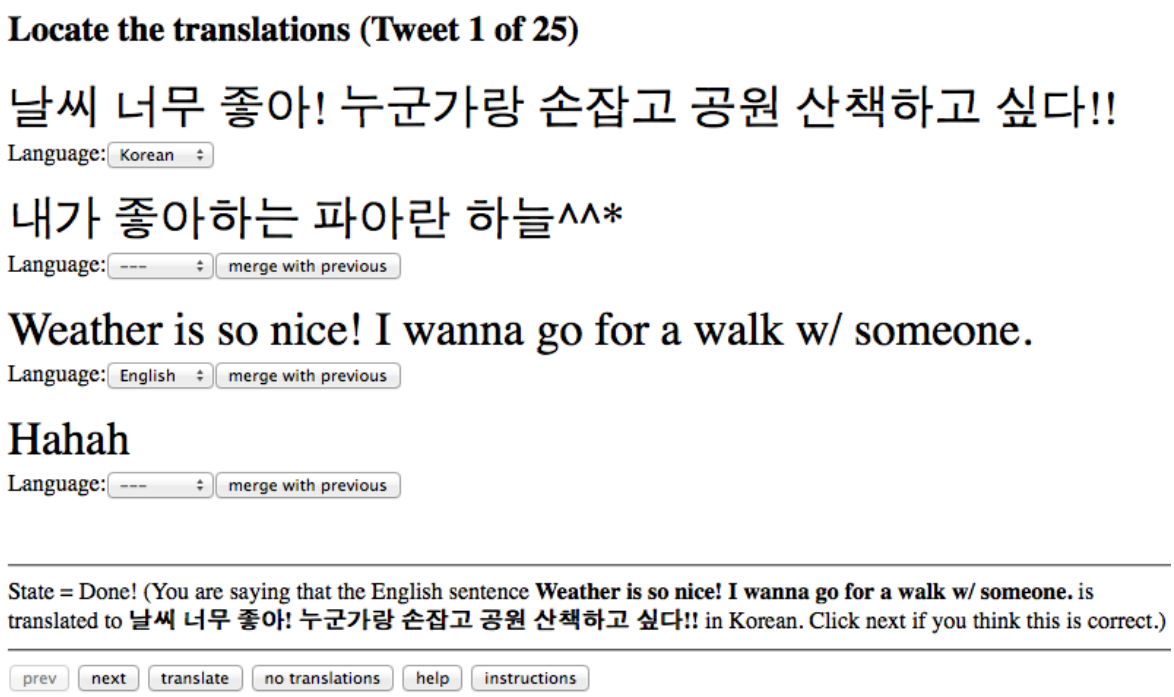

Figure 2: Location Interface (After the annotation is performed)

since he knows the reference annotation, and "lazy" because he will only define a new non-parallel segment if it is significant, otherwise it will just be left in the parallel segments. By significant, we will define whether it is at least 20% larger (in number of characters) than the parallel segments. For instance, in the Korean example in Figure 1, *Hahah* would be left in the English parallel segment, while 내가좋아하는파아란하늘 ^^* would not be in the Korean segment. We will accept a job if the average of the scores in the reference set is higher or equal than the pseudo worker's scores. This acceptor shall be denoted as $accept(lazy, a)$, where a is the number of references used.

Another option is to use the automatic system's output as a baseline that workers must improve to be accepted. We will also test this option and call this acceptor $accept(auto, a)$.

Result Combination Unlike the identification task, where the result is binary and combining multiple decisions is straightforward, the range of results from this task is larger and combining them is a challenge. Thus, we score each job based on the WER on the reference set and use annotations of the highest scoring job.

4 Experiments

To obtain results on the effectiveness of the methods described in Section 3, we will first perform experiments using pre-annotated data. We use the annotated dataset with tweets in Mandarin-English from Sina Weibo created in (Ling et al., 2013). It consists of approximately 4000 tweets crawled from Sina Weibo that were annotated on whether they contained parallel data and the location of the parallel segments. In our experiment, we sample 1000 tweets from this dataset, where 602 tweets were parallel and 398 were not.[1]

We will not submit the same tasks using different setups, since we would have to pay the cost of the tasks multiple times. Furthermore, we know the answers for all the questions in this controlled experiment, the quality of a job can be evaluated precisely by using all questions as references. Thus, we will perform the task once, with a larger number of workers and accepting and rejecting jobs based on their real quality. Then, we will use the resulting datasets and simulate the conditions using different setups.

Acceptor	$avg(a)$	$avg(r)$	d
$accept(rand, 2, 2)$	0.44	0.00	0.44
$accept(rand, 3, 4)$	0.44	0.00	0.44
$accept(rand, 4, 4)$	0.55	0.04	0.51
$accept(bal, 2, 2)$	0.69	0.09	0.60
$accept(bal, 3, 4)$	0.64	**0.03**	**0.61**
$accept(bal, 4, 4)$	**0.76**	0.15	**0.61**

Table 1: Agreement with the expert annotations for different acceptors.

4.1 Identification Task

The 1000 tweets were distributed into 40 tasks with 25 questions each ($q = 25$). Each task is to be performed by 5 workers ($n = 5$) and upon acceptance, a worker would be rewarded with 6 cents ($p = 0.06$). As we know the answers for all the questions in this case, we will calculate the Cohen's Kappa between the responses of each job and the expert annotator, and accept a job if it is higher than 0.5. We decided to use Cohen's kappa to evaluate a job, rather than accuracy, since each set of 25 questions does not contain the same number of positive and negative samples. For instance, in a set of 20 negative samples, a worker would achieve an accuracy of 80% if he simply answers negatively to all questions, which is not an adequate assessment of the job's quality. On the other hand, the Cohen's Kappa balances the positive and negative question in each task by using their prior probabilities. In total, there were 566 jobs, where 200 where accepted and 366 were rejected.

Next, we pretended that we only have access to 4 references, which will be used for quality estimation and simulate the acceptances and rejections for each strategy. Table 1 shows the averages of the real Kappa values of accepted (column $avg(a)$) and rejected jobs (column $avg(r)$) using different acceptors. Our goal is to maximize the number of acceptances with high Kappa values and minimize those that have low Kappa values. Thus, we define d as the difference between $avg(a)$ and $avg(r)$. From the results, we observe that using a balanced reference yields a much better estimation of the jobs quality using our metric d. Similar conclusions can be reached by comparing $accept(rand, 3, 4)$ with $accept(bal, 3, 4)$ and $accept(rand, 4, 4)$ with $accept(bal, 4, 4)$. Quality predictors that use balanced reference sets achieve approximately the same results for d. However, the setup $accept(bal, 3, 4)$ has a lower Kappas for both $avg(a)$ and $avg(r)$, which means that it is less likely to reject good jobs at the cost of accepting more bad jobs. This is desirable from an ethical perspective, since workers are not responsible for errors in our quality prediction. Furthermore, rejecting good jobs has a negative impact on the progress of the task, since good workers may be discouraged to perform more tasks.

[1]We wished to annotate a sample where the number of parallel posts is high, so that we would have enough samples to perform the location task.

Acceptor	prec	recall	F1	acc	κ
Automatic	0.87	0.69	0.77	0.75	0.51
All jobs	0.75	0.84	0.8	0.74	0.44
$accept(rand, 2, 2)$	0.85	0.92	0.88	0.86	0.69
$accept(rand, 3, 4)$	0.84	0.93	0.88	0.85	0.68
$accept(rand, 4, 4)$	0.91	0.95	0.93	0.92	0.82
$accept(bal, 2, 2)$	0.94	0.94	**0.94**	0.92	0.84
$accept(bal, 3, 4)$	0.93	0.95	**0.94**	**0.93**	**0.85**
$accept(bal, 4, 4)$	0.94	0.93	0.93	0.92	0.84

Table 2: Parallel post prediction scores using different acceptors.

Results on the identification task, obtained for $n = 3$, are shown in Table 2. Naturally, using a balanced reference set yields better results, since these have a higher d value. We can also see the importance of quality prediction, since not performing quality estimation (row *All jobs*) will yield worse results than the automatic system.

Next, we will compare results using different numbers of workers. We fix the quality prediction methodology to $accept(bal, 3, 4)$ and results are shown in Table 3. We observe that in general, using more workers will generate better results, but score gains from adding another worker becomes lower as n increases. One problem for $n = 2$ is the fact that there are many cases where two workers with the same weight chose a positive and a negative answer, in which case, no decision can be made, and we simply choose false by default. This explains the high recall and low precision values. However, this problem seems to occur much less with higher values of n.

4.2 Location Task

For the location task, we used the predicted parallel posts the identification task with the setup $accept(bal, 3, 4)$ and $n = 5$. We preferred to use this rather than using the expert annotations, since it would not contain false positives, which does not simulate a real situation. Then, we used 500 out of

# workers	prec	recall	F1	acc	κ
Automatic	0.87	0.69	0.77	0.75	0.51
1	0.86	0.85	0.85	0.82	0.64
2	0.85	0.95	0.90	0.87	0.72
3	0.93	0.95	0.94	0.93	0.85
4	0.94	0.96	0.95	0.94	0.87
5	0.96	0.96	**0.96**	**0.95**	**0.90**

Table 3: Identification scores for different n.

the 607 identified positive samples. This makes 20 tasks in total, with 25 questions ($q = 25$), and each task would be run until 5 jobs are accepted ($n = 5$). For this task, we set a payment of 30 cents ($p = 0.3$), since it is a more complex task. Again, since we have the expert annotations for all questions, we calculated the average WER on all answers and rejected jobs scoring less than 0.6[2].

This task is mainly focused on the quality prediction of the workers, as the result combination is done by finding the job with the highest score in the reference set. This means, for an arbitrary large n, all quality estimation methods will produce the same result, since we will find the best job on the references eventually. However, better quality estimation will allow us to find the best jobs with lower n, which makes the task less expensive. Table 4 shows results using different setups. In these results, we set aside 4 questions to be used as references. We can see that for low n (1 or 2), if we simply accept all jobs, the quality of the results will be lower than the automatic system. For $n = 4$, this approach can achieve a WER score of 0.06. However, if we use the automatic system as a baseline that jobs must surpass, we can achieve this WER score with only two jobs, which reduces the cost of this task by half. Yet, this is strongly dependent on the automatic system, as a worse system will be easier to match for the workers. On the other hand, using the smart but lazy pseudo worker, where we degrade the reference annotations slightly, we can see that we can obtain the 0.06 WER score using only the first worker. At $n = 2$, we can see that the WER improves to 0.05, which is lost for $n = 3$. This is because the prediction of the quality of the job using the workers is not always precise.

4.3 Machine Translation Results

Finally, we will perform an extrinsic test to see how the improvements obtained by using crowdsourcing map to Machine Translations. We will build an out of domain MT system using the FBIS dataset (LDC2003E14), a corpus of 300K sentence pairs from the news domain in the Chinese-English pair using the Moses (Koehn et al., 2007) pipeline. Due to the small size of our crowdsourced corpus, we will use it in the MERT tuning (Och, 2003), and test its effects compared to automatically extracted parallel data and the experts judgements. As the test set, we will use 1,500 sentence pairs from the Weibo gold standard from Ling et al. (2013), that were not used in our crowdsourcing experiment to prevent data overlap. For reordering, we use the MSD reordering model (Axelrod et al., 2005) and as the language model, we use a 5-gram model with Kneser-Ney smoothing (Heafield, 2011). Finally, results are presented with BLEU-4 (Papineni et al., 2002).

We build 3 tuning corpora, the automatically extracted corpus (denoted *Auto*), the crowdsourced corpus (denoted *Crowd*) and the corpus annotated by the expert (denoted *Expert*). This is done by taking the 1000 tweets used in this experiment, select those that were identified as parallel according to each criteria. For the automatic extraction, the authors in (Ling et al., 2013) simply use all tweets as parallel, which may influence the tuning results. Thus, we test two versions of this corpus, one where we take all samples as parallel (denoted *Auto (All)*), and one where we use the expert's decision for the identification task only (de-

[2]Determined empirically

Number of jobs	1	2	3	4	5
Automatic	0.16	0.16	0.16	0.16	0.16
All Jobs	0.23	0.21	0.07	**0.06**	**0.06**
$accept(auto, 4)$	0.09	0.06	**0.06**	**0.06**	**0.06**
$accept(lazy, 4)$	**0.06**	**0.05**	**0.06**	**0.06**	**0.06**

Table 4: Parallel data location scores for different acceptors (rows) and different numbers of workers. Each cell denotes the WER for that setup.

	Auto (Pos)	Crowd	Expert	Auto (All)
Size	483	479	483	*908*
EN-ZH	10.21	10.49	**10.51**	10.71
ZH-EN	7.59	**7.87**	7.82	8.02

Table 5: BLEU score comparison using different corpora for MERT tuning. The *Size* row denotes the number of sentences of each corpus, and the *EN-ZH* and *ZH-EN* rows denote the BLEU scores of the respective language pair and tuning dataset.

Pair	Parallel	Avg(en)	cost(I)	cost(L)	total
en-ar	1512	8.3	$35.7	$43.2	$76.2
en-zh	1302	8.7	$35.7	$37.2	$70.2
en-ja	1155	7.9	$35.7	$33.0	$68.7
en-ko	1008	7.1	$35.7	$28.8	$64.5
en-ru	798	6.3	$35.7	$22.8	$58.5
all	5775	–	$178.5	$165.0	$343.5

Table 7: AMT costs for crowdsourced corpora from Twitter.

noted *Auto (Pos)*). In the crowdsourcing case, we use the $accept(bal, 3, 4)$ setup, with $n = 5$, for the identification task and the $accept(lazy, 4)$ setup, with $n = 2$, for the location task. From the resulting parallel tweets, we also remove all tweets that were used as reference in the $accept(lazy, 4)$ quality estimator, as this would give an unfair advantage to the crowdsourced corpora.

Results are shown in Table 5, where each cell contains the average BLEU score in 5 MERT runs, using a different tuning dataset. Surprisingly, using the whole set of automatically extracted corpora actually achieves better results than using carefully selected data that are parallel. We believe that is because many non-parallel segments actually contain comparable information that can be used to improve the weights during MERT tuning. However, this does not mean that the quality of the automatically crawled corpus is better than the crowdsourced and expert annotated corpus. When using a similar number of parallel sentences, we observe that using the crowdsourced corpus yields better scores than the automatically extracted corpora, comparable to experts annotations. While results are not significantly better than automatically extracted corpora, this suggests that the crowdsourced corpora has a better overall quality than automatically extracted corpora.

5 Five Language Twitter Parallel Corpus

Now that we have established the effectiveness of our technique for extracting high-quality parallel data in a scenario where we have gold standard annotations, we apply it to creating parallel corpora in five languages on Twitter, for which we have no gold-standard parallel data: Arabic, Mandarin, Japanese, Korean and Russian. Once again, we use the extracted automatically Twitter corpus from Ling et al. (2013) and deploy the task in Mechanical Turk. We use the setup that obtained the best results in Section 4. For the identification task, we used the $accept(bal, 3, 4)$ setup, with $n = 5$. The payment for each task was 0.06 dollars. Thus, for this task, each dollar spent yields 70 annotated tweets. For the location task, we used the $accept(lazy, 4)$ setup, with $n = 2$ and each task was rewarded with 0.3 dollars. To obtain the tweet sample, we filtered the corpora in Ling et al. (2013) for tweets with alignment scores higher than 0.1. Then, we uniformly extracted 2500 tweets for each language. To generate gold standard references, the authors manually annotated 40 samples for each pair.

Table 7 contains information about the resulting corpora. The number of parallel sentences extracted from the 2500 tweets in each language pair is shown in column *Parallel* and we can see that this differs given the language pair. We can also see in column *Avg(en)* that the average number of English words is much smaller than what is seen in more formal domains. Finally, Arabic parallel data seems more predominant from our samples followed by Mandarin, while Russian parallel data seem scarcer.

6 Discriminative Parallel Data Detection

While the work in (Ling et al., 2013) used a linear combination of three models, the alignment, language and segment features, these weights were determined manually. However, using the crowdsourced corpus (in Section 5), we will apply previously proposed methods that learn a classifier with machine learning techniques as in related work on finding parallel data (Resnik and Smith, 2003; Munteanu and Marcu, 2005). In our work, we use a max entropy classifier model, similar to that presented by Munteanu and Marcu (2005) to detect parallel data in tweets. Our features are:

- **Alignment feature** - The baseline feature is the alignment score from the work in (Ling et al., 2013), and measures how well the parallel segments align, which is derived from the content-based matching methods for detecting parallel data (Resnik and Smith, 2003).

- **User features** - An observation in (Ling et al., 2013) is that a user that frequently posts in parallel is likely to post more parallel messages. Based on this, we added the average alignment score from all messages of the same user and the ratio of messages that are predicted to be parallel as features.

	Weibo (en-zh)	Twitter (en-zh)	Twitter (en-ar)	Twitter (en-ru)	Twitter (en-ko)	Twitter (en-ja)
Alignment	0.781	0.599	0.721	0.692	0.635	0.570
+User	0.814	0.598	0.721	0.705	0.650	0.566
+Length	0.839	0.603	0.725	0.706	0.650	0.569
+Repetition	**0.849**	0.652	0.763	0.729	0.655	0.579
+Language	**0.849**	**0.668**	**0.782**	**0.737**	**0.747**	**0.584**

Table 6: Classification Results using a 10-fold cross validation over different datasets. Each cell contains the F-measure using a given dataset and an incremental set of features.

- **Repetition features** - There are many words that are not translated, such as hashtags, at mentions, numbers and named entities. So, if we see these repeated twice in the same post, it can be used as a strong cue that this was the result of a translation. Hence, we define features for each of these cases, that trigger if either of these occur in multiples of two times in the same post. Named Entities were identified using a naive approach by considering words with capital letters.

- **Length feature** - It is known that the length differences between parallel sentences can be modelled by a normal distribution (Gale and Church, 1991). Hence, we used parallel data in the respective language to determine $(\tilde{\mu}, \tilde{\sigma}^2)$, which lets us calculate the likelihood of two hypothesized segments being parallel. Since we did not have annotated parallel data for this domain, we used the top 2000 scoring parallel sentences from the respective Twitter dataset in (Ling et al., 2013).

- **Language feature** - It is common for non-English words to be found in English segments, such as names of foreign celebrities, numbers and hashtags. However, when this happens to the majority of the words in a segment that is supposed to be English, it may indicated that there was an error in the language detection. The same happens with non-English segments. We used the same naive approach to detect languages as in (Ling et al., 2013), where we calculate the ratio of number of words in the English segment and the total number of words from the segment detected as English and the ratio of the number of Foreign words and the total number of words in the Foreign segment ,detected by their unicode ranges. This was also included in the work in (Ling et al., 2013).

Results using a 10 fold cross-validation are shown in Table 6. In general, we can see that the classifier performs worse in Twitter datasets compared to the Weibo dataset. We believe that this is because parallel sentences extracted from Twitter are smaller, due to the 140 character limit, which does not hold in Sina Weibo. Each parallel English segment from the Sina Weibo parallel data contains 15.4 words on average. On other hand, we see in Table 7 that this number is smaller in the parallel data from Twitter. This means that the aligner will have a much smaller range of words to align when detecting parallel data, which makes it more difficult to find parallel segments.

As for the features, we observe that by defining these simple features, we can get a significant improvement over previous baselines. For the **User** feature, we see that the improvements in the Weibo dataset are much larger than in the Twitter datasets. This is because the Twitter dataset was crawled uniformly, whereas the Weibo dataset was focused on users that post parallel data frequently. Thus, in the Weibo dataset there more posts that were posted by the same user, which does not happen as frequently in the Twitter dataset. As for the **Length** feature, we can see that it yields a small but consistent improvement over all datasets. **Repetition** based features also lead to improvements across all datasets, and produces a 5% improvement in the English-Mandarin Twitter dataset. Finally, **language** based features also add another improvement over previous results.

7 Conclusions

We presented a crowdsourcing approach to extract parallel data from tweets. As opposed to methods to crowdsource translations, our tasks do not require workers to translate sentences, but to find them in tweets. Our method is divided into two tasks. First, we identify which tweets contain translations, and we show that multiple worker's jobs can be combined to obtain results compara-

ble to those of expert annotators. Secondly, tweets that are found to contain translations are given to other workers to locate the parallel segments, where we can also obtain high quality results. Then, we use our method to extract high quality parallel data from Twitter in 5 language pairs. Finally, we improve the automatic identification of tweets with translations by using a max entropy classifier trained on the crowdsourced data.

We are currently extracting more data and the crowdsourced parallel data from Twitter will made be available to the public.

References

[Ambati and Vogel2010] Vamshi Ambati and Stephan Vogel. 2010. Can crowds build parallel corpora for machine translation systems? In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Ambati et al.2012] Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. 2012. Collaborative workflow for crowdsourcing translation. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, CSCW '12, pages 1191–1194, New York, NY, USA. ACM.

[Axelrod et al.2005] Amittai Axelrod, Ra Birch Mayne, Chris Callison-burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings International Workshop on Spoken Language Translation (IWSLT*.

[Bannard and Callison-burch2005] Colin Bannard and Chris Callison-burch. 2005. Paraphrasing with bilingual parallel corpora. In *In ACL-2005*, pages 597–604.

[Bojar et al.2013] Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

[Das and Petrov2011] Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 600–609, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Gale and Church1991] William A. Gale and Kenneth W. Church. 1991. A program for aligning sentences in bilingual corpora. In *Proceedings of the 29th Annual Meeting on Association for Computational Linguistics*, ACL '91, pages 177–184, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Gale et al.1992] William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. Using bilingual materials to develop word sense disambiguation methods.

[Ganitkevitch et al.2012] Juri Ganitkevitch, Yuan Cao, Jonathan Weese, Matt Post, and Chris Callison-Burch. 2012. Joshua 4.0: Packing, PRO, and paraphrases. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 283–291, Montréal, Canada, June. Association for Computational Linguistics.

[Goutte et al.2012] Cyril Goutte, Marine Carpuat, and George Foster. 2012. The impact of sentence alignment errors on phrase-based machine translation performance. In *Proc. of AMTA*.

[Heafield2011] Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, July.

[Jehl et al.2012] Laura Jehl, Felix Hieber, and Stefan Riezler. 2012. Twitter translation using translation-based cross-lingual retrieval. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 410–421, Montréal, Canada, June. Association for Computational Linguistics.

[Koehn et al.2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-burch, Richard Zens, Rwth Aachen, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondrej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

[Ling et al.2013] Wang Ling, Guang Xiang, Chris Dyer, Alan Black, and Isabel Trancoso. 2013. Microblogs as parallel corpora. In *Proceedings of the 51st Annual Meeting on Association for Computational Linguistics*, ACL '13. Association for Computational Linguistics.

[Liu et al.2011] Feifan Liu, Fei Liu, and Yang Liu. 2011. Learning from chinese-english parallel data for chinese tense prediction. In *IJCNLP*, pages 1116–1124.

[Munteanu and Marcu2005] Dragos Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting comparable corpora. *Computational Linguistics*, 31(4):477–504.

[Ng et al.2003] Hwee Tou Ng, Bin Wang, and Yee Seng Chan. 2003. Exploiting parallel texts for word sense disambiguation: An empirical study. In *Proceedings of ACL03*, pages 455–462.

[Och2003] Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Papineni et al.2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Post et al.2012] Matt Post, Chris Callison-Burch, and Miles Osborne. 2012. Constructing parallel corpora for six indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada, June. Association for Computational Linguistics.

[Resnik and Smith2003] Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29:349–380.

[Schwarck et al.2010] Florian Schwarck, Alexander Fraser, and Hinrich Schütze. 2010. Bitext-based resolution of german subject-object ambiguities. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 737–740, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Smith et al.2010] Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 403–411, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Specia et al.2005] Lucia Specia, Maria Das Graças, Volpe Nunes, and Mark Stevenson. 2005. Exploiting parallel texts to produce a multilingual sense tagged corpus for word sense disambiguation. In *Proceedings of RANLP-05, Borovets*, pages 525–531.

[Zaidan and Callison-Burch2011] Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: professional quality from non-professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1220–1229, Stroudsburg, PA, USA. Association for Computational Linguistics.