Mining Parallel Corpora from Sina Weibo and Twitter

Wang Ling*
Google DeepMind

Luís Marujo**
Feedzai Research

Chris Dyer†
Carnegie Mellon University

Alan W. Black‡
Carnegie Mellon University

Isabel Trancoso§
Instituto Superior Técnico
University of Lisbon
INESC-ID

Microblogs such as Twitter, Facebook, and Sina Weibo (China's equivalent of Twitter) are a remarkable linguistic resource. In contrast to content from edited genres such as newswire, microblogs contain discussions of virtually every topic by numerous individuals in different languages and dialects and in different styles. In this work, we show that some microblog users post "self-translated" messages targeting audiences who speak different languages, either by writing the same message in multiple languages or by retweeting translations of their original posts in a second language. We introduce a method for finding and extracting this naturally occurring parallel data. Identifying the parallel content requires solving an alignment problem, and we give an optimally efficient dynamic programming algorithm for this. Using our method, we extract nearly 3M Chinese–English parallel segments from Sina Weibo using a targeted crawl of Weibo users who post in multiple languages. Additionally, from a random sample of Twitter, we obtain substantial amounts of parallel data in multiple language pairs. Evaluation is performed by assessing the accuracy of our extraction approach relative to a manual annotation as well as

* Google DeepMind, London, N1 0AE. E-mail: lingwang@google.com.
** Feedzai Research, Lisbon, 1990-095. E-mail: luis.marujo@feedzai.com.
† School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. E-mail: cdyer@cs.cmu.edu.
‡ School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. E-mail: awb@cs.cmu.edu.
§ Instituto Superior Técnico, Lisbon, 1000-029. E-mail: isabel.trancoso@inesc-id.pt.

Submission received: 10 July 2014; accepted for publication: 25 January 2016.

doi:10.1162/COLI_a_00249

in terms of utility as training data for a Chinese–English machine translation system. Relative to traditional parallel data resources, the automatically extracted parallel data yield substantial translation quality improvements in translating microblog text and modest improvements in translating edited news content.

1. Introduction

In the span of about two decades, the Web has evolved from a collection of mostly static Web pages created, to dynamic, interactive content, and to content created by users themselves. The advent of microblogs, such as Facebook and Twitter, has particularly transformed the kind of information that is published, since traditional barriers to publication (e.g., expertise in Web development) have been virtually eliminated by making publication easy for anyone with Web access. Microblog posts are created in the service of a variety of communicative and identity-building goals (Marwick and Boyd 2010), and the diversity of users and their intentions is reflected in an equally diverse array of writing styles and linguistic conventions. In contrast to traditional print media, user-generated content on social media can be informal, colloquial, and is in particular marked by innovative and varied use of orthography. For example, we can readily find tweets like (*R U still with me or what?*) and nonstandard abbreviations (*idk! smh*).

Automated language processing tools (e.g., those that perform linguistic analysis or translation) face particular difficulty with this new kind of content. On one hand, these have been developed with the conventions of more edited genres in mind. For example, they often make strong assumptions about orthographic and lexical uniformity (e.g., that there is just one way to spell *you*, and that *cool*, *cooool*, and *cooooool* represent completely unrelated lexical items). While modeling innovations are helping to relax these assumptions (Han and Baldwin 2011; Ritter et al. 2012; Owoputi et al. 2013; Ling et al. 2015), a second serious challenge is that many of the annotated text resources that our tools are learned from are drawn from edited genres, and poor generalization from edited to user-generated genres is a major source of errors (Gimpel et al. 2011; Kong et al. 2014).

In this work, we present methods for finding naturally occurring parallel data on social media sites that is suitable for training machine translation (MT) systems. In MT, the domain mismatch problem is quite acute because most existing sources of parallel data are governmental, religious, or commercial, which are quite different from user-generated content, both in language use and in topic. The extracted parallel data can then be used to create systems designed to translate user-generated content. Additionally, because microblogs host discussions of virtually limitless topics, they are also a potential source of information about how to translate names and words associated with breaking events, and, as such, may be useful for translation of texts from more traditional domains. Apart from machine translation, parallel data in this domain can improve and help create applications in other areas in NLP (Ling et al. 2013; Peng, Wang, and Dredze 2014; Wang et al. 2014).

Our method is inspired by the (perhaps surprising) observation that a reasonable number of microblog users tweet "in parallel" in two or more languages. For instance, the American entertainer Snoop Dogg maintains an account on Sina Weibo that regularly posts English–Chinese parallel messages, for example, *watup Kenny Mayne!! - Kenny Mayne* 最近这么样啊!!, where an English message and its Chinese translation are in the same post, separated by a dash. It is not only celebrities (or

their publicists) who tweet in multiple languages; we see this with casual users as well. For example, on Twitter, we found こんな方きは嫌だ | *I don't like this Ehomaki!!: http://t.co/9FTXgV0w*, which contains a Japanese to English translation, separated by | and followed by an image link, which is displayed as an image when viewed on the Twitter Web site. Other examples of parallel posts are shown in Figure 1. We note that these parallel posts contain information that is valuable, since they contain elements that are rare in standard edited genres. In the examples we have given here, the terms *watup* and *Ehomaki* are not correctly or not translated by online MT systems.

We offer brief speculation on the reasons that users translate their own posts, although later in the article we will provide an analysis of the content that provides further insight into the question. One class of parallel posts are found in public celebrities' profiles (e.g., Snoop Dogg's Sina Weibo posts), done to enhance their popularity in non-English markets. Another class is posted by people who live or have lived abroad and wish to communicate with people who speak different languages. In fact, our first contact with such posts was with the parallel messages that Chinese students at Carnegie Mellon University were posting on Facebook so their friends both in China and in Pittsburgh could read them.

Extracting parallel data from such posts poses many challenges to current NLP and MT methods. As part of the elaboration of this work, we made the following contributions to the field of NLP and MT.

Domain	Lang	Tweet
Twitter	DE-EN	RT @fcbayern_news: Nur noch 24 Stunden / Only 24 hours remaining #finaledahoam #fcb
Twitter	FR-EN	Qui est le véritable avare ? Who is the real miser?
Twitter	ES-EN	YEAH YEAH Im mexican so i should tweet in Spanish I KNOW SO I WILL.VSI SI SI soy mexicana así que debería twitear en español LO SE Y LO HARÉ
Twitter	PT-EN	Ótimo lugar pra visitar na Alemanha: Dresden!\n- ~ -\nGreat place to visit in Germany: Dresden!
Twitter	AR-EN	؟□ .. التويتر في مشكلتي نفس عنده من Who has the same of my problem .. □? http://t.co/hIrJdKOt
Twitter	JA-EN	オーバーヘッドプロジェクタと共に使われる透明画 - a transparency for use with an overhead projector
Twitter	KO-EN	날씨 너무 좋아! 누군가랑 손잡고 공원 산책하고 싶다!! 내가 좋아하는 파아란 하늘^^*Weather is so nice! I wanna go for a walk w/ someone. Hahah
Twitter	RU-EN	http://t.co/Yz6qmhHV меня никто не спрашивает, он просто ДОЛЖЕН стать моим любимым оппой <3 | He MUST to be my lovely oppa! <3
Twitter	ZH-EN	奥巴马公开宣称支持同性恋婚姻 Barack Obama speaks out and declares support for same-sex marriage http://t.co/gle6PKJG 副总统拜登道歉拖奥黑下水 http://t.co/tPVmaFWW
Weibo	ZH-AR	可爱熟睡婴儿الرضيع النائم المحبوب http:\/\/t.cn\/zOL7Xr1
Weibo	DE-ZH	Ich bin allein , aber doch nicht allein . （我是一个人，但我并不孤单） [亲亲]
Weibo	PT-ZH	Precisamos da ajuda de todos os torcedores n os empurrando muito na partida de hj!vamos para cima deles.Forca Yiteng!!球迷们,今天的比赛非常需要你们的助威呐喊,我们共同来争取胜利.加油,强大的毅腾！！

Figure 1
Examples of parallel posts in different language pairs and from different sources. Translated material is highlighted, non-translated material is not. DE = German; EN = English; ES = Spanish; PT = Portuguese; AR = Arabic; JA = Japanese; KO = Korean; RU = Russian; ZH = Chinese.

- **A method for extracting parallel data within documents** - Existing methods for detecting parallel data (Resnik and Smith 2003; Fukushima, Taura, and Chikayama 2006; Li and Liu 2008; Uszkoreit et al. 2010; Ture and Lin 2012) assume that the source and target documents are separate. That is, these methods reason about the probability that documents *A* and *B* are translations. Previous work on extracting parallel data from Twitter (Jehl, Hieber, and Riezler 2012) retrieves candidate pairs of tweets and determines whether the tweets in the pair are translations. In our work, single tweets are considered, and the problem is to determine whether the tweet contains translations and, if so, which spans are parallel.
- **Efficient multilingual document detection** - Parallel tweets are a relative rarity, but because of the volumes of data involved, we can find a large amount of content. We must efficiently detect multilingual tweets, as monolingual tweets do not contain translations. Although language identification is a well-studied problem (Zissman 1996; Gottron and Lipka 2010; Greenhill 2011), even in the microblog domain (Bergsma et al. 2012), these assume that only one language is present within a document and cannot be directly applied to this problem. Furthermore, because of the magnitude of tweets that must be processed, many of the proposed solutions cannot be applied due to their computational complexity. In our work, we propose an efficient implementation for large-scale detection of multilingual documents.
- **Crowdsourcing for parallel data extraction** - To tune the parameters of our parallel data extractor and perform MT experiments, user-verified annotations must be obtained. To obtain this, we propose a simple crowdsourcing method for extracting parallel data from tweets.

At a high level, our algorithm proceeds in three steps. First, we identify candidate multilingual tweets using a fast heuristic classifier. Second, we extract the candidate parallel segments from each tweet using a dynamic programming algorithm. Third, a classifier determines if the proposed parallel segment is correct. If so, the material is extracted, otherwise it is discarded.

This research is an extension of the preliminary work described in Ling et al. (2013), in which we obtained over 1 million Chinese–English parallel segments from Sina Weibo, using only their public application program interface (API). This automatically extracted parallel data yielded substantial translation quality improvements in translating microblog text and modest improvements in translating edited news. Following this work, we developed a method for crowdsourcing judgments about parallel segments (Ling et al. 2014), which was then used to build gold standard data for other language pairs and for the Twitter domain. This article extends these two papers in several ways:

- **Improved language pair detection** - The previous work assumes that the language pair is formed by two languages with different unicode ranges, such as English–Chinese, and does not support the extraction of parallel data if the languages share the same unicode range (such as English–Portuguese). This issue is addressed in this article, where we present a novel approach for finding multilingual tweets.

- **More language pairs considered** - The previous architecture only allowed one language pair to be considered during extraction. Thus, for instance, only English–Chinese parallel sentences were extracted from Sina Weibo, and English–Arabic sentence pairs from Twitter. In this work, we present a general architecture that allows multiple language pairs to be considered simultaneously. Using this architecture, we extracted data for nine language pairs from Twitter.[1]
- **MT experiments with more data** - As we are actively collecting more data, our resulting parallel data sets are naturally larger. In fact, our current data set amounts to nearly 3 million sentence pairs for the English–Chinese language pair alone. Furthermore, we perform cross-domain experiments (training using Weibo data and testing on Twitter data) and consider other language pairs.

The article is organized as follows. Section 2 describes the related work in parallel data extraction. Section 3 presents our model to align segments within one single document and the metrics used to evaluate the quality of the segments. Section 4 describes the extraction pipeline used to extract the parallel data from Sina Weibo and Twitter. Section 5 describes the method used to obtain gold standards for translation in different languages. We then present, in Section 7, the experiments showing that our harvested data not only substantially improve translations of microblog text with existing (and arguably inappropriate) translation models, but that they improve the translation of more traditional MT genres, such as newswire. We conclude in Section 8.

2. Related Work

The automatic identification and retrieval of translated text (**bitext**) is a well-studied problem in natural language processing. In particular, there has been a long tradition of exploiting the multilingual Web as a source of bitext (Resnik and Smith 2003; Fukushima, Taura, and Chikayama 2006; Li and Liu 2008; Uszkoreit et al. 2010; Ture and Lin 2012). In general, existing approaches extract bitext in two steps.

In the first step, candidate pairs of Web sites that are likely to be parallel are retrieved from a set of documents using an efficient retrieval mechanism. This step is necessary to narrow down the potential candidates for parallel Web sites, since considering all possible pairs in a given set of Web pages is intractable (a direct comparison of $O(n^2)$ documents would be impractical for any modestly sized collection). Resnik and Smith (2003) used features from URLs to build this set of pairs, by checking for Web sites with patterns that may indicate the presence of a parallel Web site. For instance, the pattern *lang=en* generally indicates that by changing *en* to another language, such as *pt*, we may find another Web site, which is the translation of the original one. However, this method has a low recall, because it ignores the content within the Web document. Uszkoreit et al. (2010) proposed a content-based approach to improve retrieval that executes this first step by translating all documents into English using a baseline translation system, and then finds rare (i.e., informative) n-grams that are common across Web documents with different original languages using an inverted indexing approach.

1 We also extracted parallel sentence pairs translated from Chinese into nine other languages, which are also available, but since no MT experiments were conducted, these are not described here.

The second step determines whether a candidate pair of documents is actually a translation pair by examining the content of the documents. Resnik and Smith (2003) proposed structural filtering, a language independent approach, which compares the HTML structures of the pair of documents, as well as a content-based approach. One possible implementation is to word-align (Brown et al. 1993) the documents using an existing lexical translation model, and then compute the following score:

$$S = \frac{\#\text{alignments}}{\#\text{alignments} + \#\text{unaligned words}} \tag{1}$$

Finally, a threshold is set and pairs of Web pages are considered parallel if their score is higher than the threshold.

One approach to obtain word alignments is IBM Model 1, which we review now because our method will also depend on it. Given a source sentence $\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle$ and a target translation $\mathbf{y} = \langle y_1, y_2, \ldots, y_m \rangle$, where n and m denote the number of tokens in $\mathbf{x}$ and $\mathbf{y}$, IBM Model 1 defines the joint probability of a lexical alignment $\mathbf{a}$ and translation $\mathbf{y}$ given input $\mathbf{x}$ as

$$P_{M1}(\mathbf{a}, \mathbf{y} \mid \mathbf{x}) = \frac{1}{(n+1)^m} \prod_{i=1}^{m} t(y_i \mid x_{a_i}) \tag{2}$$

where $\mathbf{a} \in [0, n]^m$ gives the alignment of each word in $\mathbf{y}$ to a word in $\mathbf{x}$ (or a null word, indicating it is not a translation), and $t(y_i \mid x_{a_i})$ denotes the lexical translation probability that word y_i is the translation of x_{a_i}. Model 1 naively assigns a uniform prior probability to all alignment configurations. Although this is an obviously flawed assumption, the posterior alignment probability under Model 1 (i.e., $P_{M1}(\mathbf{a} \mid \mathbf{x}, \mathbf{y})$) is surprisingly informative. More robust models make less-naive prior assumptions and generally produce higher-quality alignments, but the uniform prior probability assumption simplifies the complexity of performing inference. Despite its simplicity, Model 1 has shown particularly good performance as a component in sentence alignment systems (Xu, Zens, and Ney 2005; Braune and Fraser 2010).

Some work on parallel data extraction has also focused on extracting parallel segments from comparable corpora (Smith, Quirk, and Toutanova 2010; Munteanu, Fraser, and Marcu 2004). Smith et al. (2010) uses conditional random fields to identify parallel segments from comparable Wikipedia documents (since Wikipedia documents in multiple languages are not generally translations of each other, although they are about the same topics). The work of Jehl, Hieber, and Riezler (2012) uses cross-lingual information retrieval techniques to extract candidate English–Arabic translations from Twitter. These candidates are then refined using a more expressive model to identify translations (Xu, Weischedel, and Nguyen 2001).

It is also possible to focus the extraction in one particular type of phenomena. For example, the work on mining parenthetical translations (Lin et al. 2008), which attempts to find translations within the same document, has some similarities with our work, since parenthetical translations are within the same document. However, parenthetical translations are generally used to translate names or terms, which is more limited than our work targeting the extraction of whole sentence translations. More recently, a similar method for extracting parallel data from multilingual Facebook posts was proposed (Eck et al. 2014).

Finally, crowdsourcing techniques can be used to obtain translations of text from new domains (Ambati and Vogel 2010; Zaidan and Callison-Burch 2011; Ambati, Vogel, and Carbonell 2012; Post, Callison-Burch, and Osborne 2012). These approaches require compensating workers for their efforts, and the workers themselves must be generally proficient in two languages, making the technique quite expensive. Previous work has relied on employing workers to translate segments. Crowdsourcing methods must also address the need for quality control. Thus, in order to find good translations, subsequent post-editing and/or ranking is generally necessary.

3. The Intra-Document Alignment (IDA) Model

As discussed above, content-based filtering is a method for parallel data detection that relies on lexical translation/alignment models to provide evidence that a candidate pair is actually a translation. However, in previous work, the candidate pairs have been independent documents. In contrast, in our problem, translated material is embedded in a single document, with different regions in different languages. In this section, we describe a model for identifying translated material in these documents.

The intra-document alignment (IDA) model takes as input a document $\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle$, where n denotes the number of tokens in the document. Our goal is to find the location of the parallel segments, the languages of those segments, and their word alignments (for simplicity, we assume that there are at most two continuous segments that are parallel). These are specified by the tuple $([p, q], l, [u, v], r, \mathbf{a})$, where the token indexes $[p, q]$ and $[u, v]$ are used to identify the left segment (from p to q) and right segment (from u to v), which are parallel. We shall refer to $[p, q]$ and $[u, v]$ as the **spans** of the left and right segments. To avoid degenerate solutions due to overlapping content, we require that $p \leq q < u \leq v$. We use l and r to identify the language of the left and right segments, respectively. Finally, $\mathbf{a}$ represents the word alignment between the words in the left and the right segments. An example analysis is shown in Figure 2. In the given tweet, the left and right segment spans $[1, 5]$ and $[11, 13]$ are identified in (A), the languages for these spans *zh* and *en* in (B), and the alignments $(2, 12), (3, 13)$, indicating that words in indexes 2 and 3 are aligned to words at indexes 12 and 13, are shown in (C).

In the following sections, we will present these three components (Section 3.1) and a dynamic programming search algorithm for finding the best analysis of a tweet under the model (Section 3.2). The basic version of the algorithm is tuned to detect parallel data in a particular pair of languages, and in Section 3.3, we discuss an extension of algorithm to deal with more than a single language pair. Finally, we will describe the metrics used to evaluate the quality of the extracted parallel segments in Section 3.4.

3.1 Model Components

The core of our parallel span detection model is IBM Model 1 (whose parameters are learned from seed parallel data). This model assigns a conditional probabilities to a string of words in one language given a string of words in the other language. Although this model will assign higher probabilities to translated content, there are several problems that must be dealt with. First, there is a bias for selecting shorter spans, since longer spans must consist of more (conditionally independent) generation events whose probabilities multiply. To understand the problem, consider the example in Figure 2, and assume that we know the language pair l, r and the word alignments a.

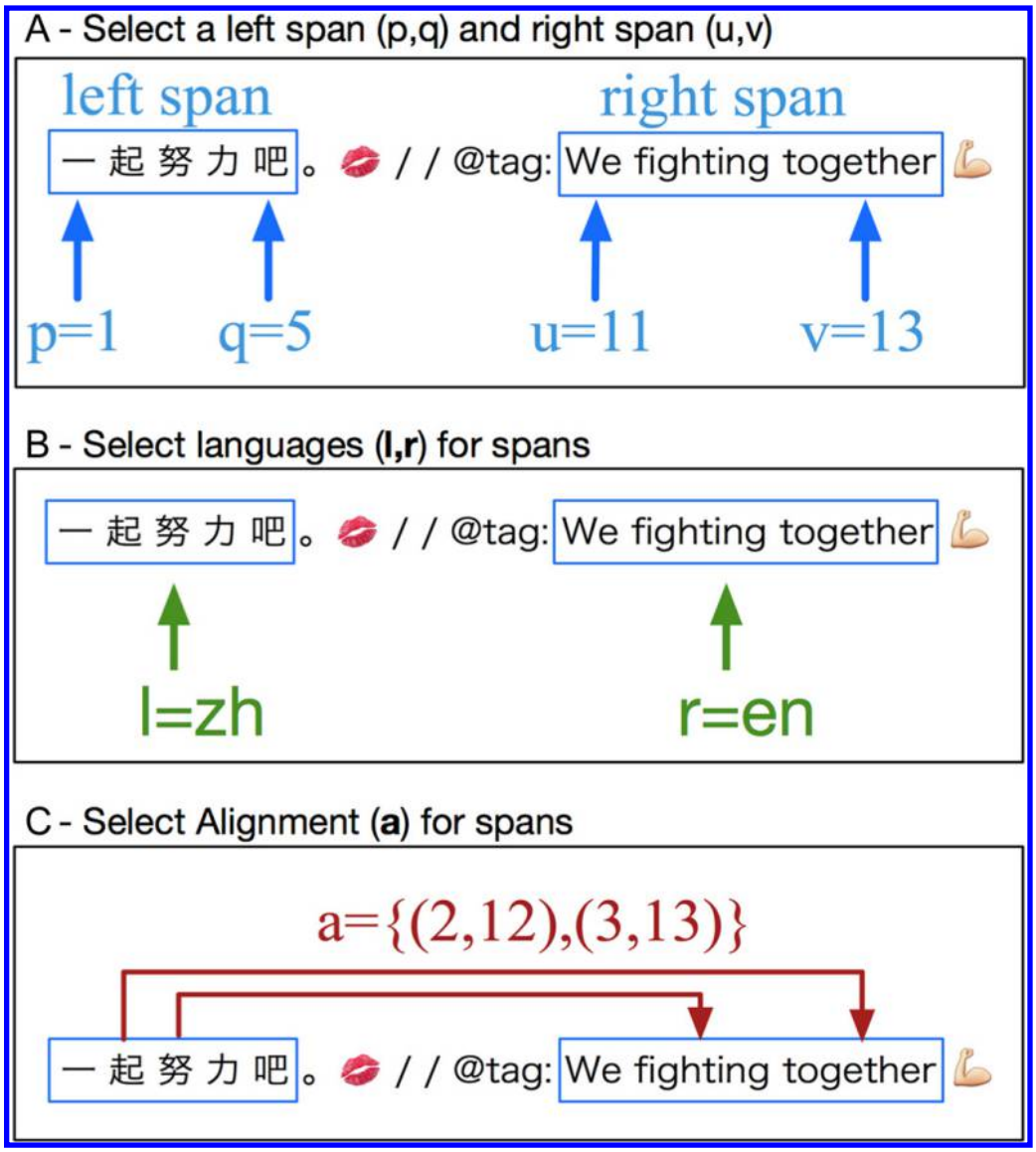

Figure 2
Illustration of each of the model parameters. (A) A potential pair of parallel segments p, q, u, v. (B) A possible pair of languages l, r. (C) The word alignments between these segments.

Suppose that the Chinese characters 起 and 努 are aligned to the words *fighting* and *together*, respectively. We now consider three possible segment spans $[p, q], [u, v]$: (1) The desired translation from 一起努力吧 to *We fighting together*; (2) the segment of all aligned words translating from 起努 to *fighting together*; (3) and the segment with a word-to-word translation from 起 to *together*. According to Equation (2), the translation probabilities would be calculated as:

- (1) $\frac{1}{4^6} P_{\text{M1}}(\textit{together} \mid 起) P_{\text{M1}}(\textit{fighting} \mid 努)$

- (2) $\frac{1}{2^3} P_{\text{M1}}(\textit{together} \mid 起) P_{\text{M1}}(\textit{fighting} \mid 努)$
- (3) $\frac{1}{1^3} P_{\text{M1}}(\textit{together} \mid 起)$

A second problem is that the model is unlikely to choose segments with unaligned words. This can be seen by comparing the first and second cases, where it is evident that the first case's translation probability is bounded by the second one regardless of the word translation probabilities, as the normalization factor $\frac{1}{n^{m+1}}$ is inversely proportional to the number of words in the segments. A more aggravating issue is the fact that the model will always pick word-to-word translation segments. This can be seen by comparing case (2) and (3), where we can observe that the second case is bounded by the third one, as the second case is a product of more translation probabilities.

Similar results are obtained if we consider Equation (1), where the example (1) obtains a lower score than (2) and (3), because it contains unaligned words.

To address this problem, we propose a three-feature model that takes into account the spans of the parallel segments, their languages, and word alignments. This model is defined as follows:

$$\textit{score}([u,v],r,[p,q],l,\mathbf{a} \mid \mathbf{x}) = \underbrace{\phi_S([p,q],[u,v] \mid l,r,\mathbf{x})}_{\text{span score}} \times \underbrace{\phi_L(l,r \mid [p,q],[u,v],\mathbf{x})}_{\text{language score}} \times \underbrace{\phi_T(\mathbf{a} \mid [p,q],l,[u,v],r,\mathbf{x})}_{\text{translation score}}$$

Each of the component scores (ϕ_S, ϕ_L, and ϕ_T) returns a score in the interval $[0,1]$. We now describe the role each score plays in the model.

3.1.1 Translation Score. The translation score $\phi_T(\mathbf{a} \mid [p,q],l,[u,v],r)$ indicates whether $[p,q]$ is a reasonable translation of $[u,v]$, according to an arbitrary alignment $\mathbf{a}$. Previously, we relied on IBM Model 1 probabilities for this score:

$$\phi_{M1}(\mathbf{a} \mid [p,q],l,[u,v],r,\mathbf{x}) = \frac{1}{(q-p+2)^{v-u+1}} \prod_{i=u}^{v} t_{\text{M1}}^{l \to r}(x_i \mid x_{a_i})$$

This equation is a reformulation of Model 1 presented in Equation 2 for a single document with the segment p,q as source, and u,v as target. The lexical tables $t_{\text{M1}}^{l \to r}$ for the various language pairs are trained a priori using available parallel corpora. The null translation probability is set to ϵ, which is set to a negligible probability, so that these are only used if x_i cannot be aligned to any word. Note that the translation score by itself also allows the identification of the language pair l,r, as using a lexical table from an incorrect language pair is likely to yield lower scores.

However, when considering multiple language pairs, lexical translation probabilities for pairs of words in some language pairs tend to be higher, as there are fewer equivalent translations for the same word. This would bias the model to pick those language pairs more often, which is not desirable. Thus, we redefine this score by adapting the content-based matching metric, which assumes lexical translation uniformity:

$$\phi_{\textit{match}}(\mathbf{a} \mid [p,q],l,[u,v],r,\mathbf{x}) = \frac{\#\mathbf{a}}{\sum_{i \in [p,q] \cup [u,v]} \delta(i \notin a)}$$

However, the computation of the alignments $\mathbf{a}$ for a given language pair l, r is still performed under IBM Model 1.

Finally, as performing word alignments for different directions can generate a different set of alignments, we also compute the alignments from r to l. These alignments are generated for both directions, which are denoted as $\mathbf{a}_{l,r}$ and $\mathbf{a}_{r,l}$. The translation score is defined as:

$$\phi_T = \max(\phi_{match}(\mathbf{a}_{l,r} \mid [p,q], l, [u,v], r, \mathbf{x}), \phi_{match}(\mathbf{a}_{r,l} \mid [u,v], r, [p,q], l, \mathbf{x}))$$

3.1.2 Span Score. The span score $\phi_S([p,q],[u,v] \mid l, r, \mathbf{x})$ of hypothesized pair of segment spans $[p,q],[u,v]$ is defined as:

$$\phi_S(l, r \mid [p,q],[u,v] \mid \mathbf{x}) = \frac{(q-p+1)+(v-u+1)}{\sum_{0<p' \leq q' < u' \leq v' \leq n}(q'-p'+1)+(v'-u'+1)} \times \delta([p,q],[u,v] \in \omega(l,r,\mathbf{x}))$$

The span score is a distribution over all spans that assigns higher probability to segmentations that cover more words in the document. It assigns the highest probability to segmentations that cover all the words in the document, which encourages the model to favor larger segments.

It also depends on the function δ, which takes the values 1 or 0, depending on whether all the segment boundaries are valid according to the list ω of violations. The only exception occurs when there are no valid boundaries, in which case δ returns 1. This violation list allows the definition of hard constraints regarding a segment's validity based on prior knowledge.

The first intuition that we wish to capture is that consecutive words in the same unicode range tend to belong to the same sentence. For instance, in the example in Figure 2, segments *fighting together* or *fighting* would not be valid, because they do not include the whole sequence of Latin words *We fighting together*. However, the segment *tag: We fighting together* or any other segment that contains the whole sequence of Latin words is acceptable. The same applies to the Chinese segment 一起努力吧.

The second rule ensures that a segment containing a parenthesis starter will also contain the corresponding parenthesis ender. For instance, for the tweet *Yoona taking the* '身体健康' *(be healthy)* ^^, the segment *(be healthy* would not be valid, because it does not contain the parenthesis closer *)*. However, both *(be healthy)* and *be healthy* are acceptable. The exception to this rule occurs when either the parenthesis starter or ender is missing in the tweet, in which case this violation is ignored. We consider the following universal starter and ender pairs: *()*, *[]*, and {}; as well as the Chinese and Japanese pairs: （）, 【】, 「」, and 『』.

With the inclusion of the span score, the model can successfully address the example in Figure 2, as the span score will give preference to longer segments and the list of violations will force the model to use complete sentences. Consider the French–English example in Figure 1, and assume that the alignment model is able to align *est* to *is*, *le* to *the* and the left question mark *?* is aligned to the right one. In this case, the current model would be able to find the correct segments since the list of violations would enforce that the left segment contains *Qui est le véritable avare* and the right segment contains *Who is the real miser?*. Furthermore, as the question marks are aligned, the model is capable

of correctly extracting the translation from *Qui est le véritable avare?* to *Who is the real miser?*.

Although the span score captures many desirable properties of good alignments based on structural features like parentheses, such features are not always present. For instance, consider that the previous example was tweeted as *Who is the real miser Qui est le véritable avare*. We can see that correctly translated segments from *Who is the real miser* to *Qui est le véritable avare* and the incorrect segments that translate from *Who is the real* to *miser Qui est le véritable avare* would receive the same score according to the span score, because they cover all words in the tweet. This is because the model does not know whether *miser* belongs to the English or French segment. One solution to solve this would be to augment the lexical table so that the alignment model can align the word *miser* to *avare*, so that the translation score is higher in the first case. However, this is an unreasonable requirement as it would require large amounts of parallel data. An easier solution would include a language detector in order to identify that it is likely that *miser* is an English word, so it is unlikely that it will be placed in the French segment. This information is encoded in the language score.

3.1.3 Language Score. The language score $\phi_L(l, r \mid [p, q], [u, v], \mathbf{x})$ indicates whether the language labels l, r are appropriate to the document contents:

$$\phi_L(l, r \mid [p, q], [u, v], \mathbf{x}) = \frac{\sum_{i=p}^{q} P_L(l \mid x_i) + \sum_{i=u}^{v} P_L(x_i \mid r)}{(q - p + 1) + (v - u + 1)}$$

where $P_L(x \mid l)$ is a language detection function that yields the probability that word x_i is in language l. In our previous work (Ling et al. 2013), this was a 0-1 function based on unicode ranges, which assumed that all Latin words were English, all Han Characters were Chinese, and all Perso-Arabic letters were Arabic. This was not a problem, because we were only extracting Arabic–English and Chinese–English pairs. To cope with any language pair—including pairs where both languages have the same unicode range—we estimate $P_L(x \mid l)$ by training a character-based language detector[2] and calculating the posterior probability of a language l given a word x.

The score benefits the model in two aspects. First, it addresses the problem mentioned above, so that we can address tweets with languages in the same unicode range. Second, it allows a better identification of the language pair l, r than simply using the alignment model. The reason for this is the fact that some languages, such as Portuguese and Spanish or Chinese and Japanese, contain overlapping words. For instance, in the tweet よい 末を！*Have a nice weekend! #ohayo*, the characters 末 mean *weekend* in both Chinese and Japanese. Thus, if these are the only characters that are aligned during the computation of the translation score, both languages are equally likely to be correct. However, the Hiragana characters よ, い, and を that surround it only exist in Japanese, so the entire sequence よい末を！ is much more likely to be Japanese than Chinese. These details are captured by the character-based language score.

2 `http://code.google.com/p/language-detection/`.

3.2 Searching for the Best Analysis

In our search problem, we need to efficiently search over all pairs of spans, languages, and word alignments. We will show that (assuming a Model 1 alignment model), a dynamic programming search algorithm is available. Our search problem is formalized as follows:

$$\big([p,q],l,[u,v],r\big)^* = \arg \max_{[p,q],l,[u,v],r} \max_{\mathbf{a}} score([p,q],l,[u,v],r,\mathbf{a} \mid \mathbf{x}) \tag{3}$$

with variables defined as in the previous section. A high model score indicates that the predicted bispan $[p,q],[u,v]$ is likely to correspond to a valid parallel span.

A naive approach to solving this problem would require each of the scores ϕ_S, ϕ_L, and ϕ_T to be computed for each possible value of l,r,p,q,u,v. Calculating the span score ϕ_S for all combinations of u,v,p,q values only requires a sum of boundary positions. The set of violations ω only needs to be computed once for each sample.

As for the language score ϕ_L, the language probabilities for each word $P_L(x \mid l)$ only need to be computed once. The detector can compute the language probabilities for a word x for all considered languages in linear time over the length of x. Furthermore, the number of character n-grams that need to be considered is relatively small, as we are applying it on words rather than documents. Once $P_L(x \mid l)$ is computed for all possible values of l and x, calculating the language score can be trivially computed.

However, computing the translation score ϕ_T requires the computation of the word alignments in ϕ_T over all possible segmentations, which requires $O(|\mathbf{x}|^6)$ operations using a naive approach [$O(|\mathbf{x}|^4)$ segmentation candidates, and $O(|\mathbf{x}|^2)$ operations for aligning each pair of segments]. Even if only one language pair l,r is considered, a document with 40 tokens would require approximately 7 million operations in the worst case for ϕ_T to be computed for all segmentations. To process millions of documents, this process needs to be optimized.

Fortunately, we can reduce the order of magnitude of this algorithm to $O(|\mathbf{x}|^4)$ without resorting to approximations. This is done by showing that, under Model 1, Viterbi alignments do not need to be recomputed every time the segment spans are changed, and can be obtained by updating previously computed ones. Thus, we propose an iterative approach to compute the Viterbi word alignments for IBM Model 1 using dynamic programming.

Finally, computing the alignments using all possible values of l,r is unnecessarily expensive, and we will show that we can limit the number of pairs without approximations by using the language identification approach described in Section 3.3.

Dynamic programming search. We leverage the fact that the Viterbi word alignment of a bispan (or pair of spans) under Model 1 can be reused to calculate the Viterbi word alignments of larger bispans. The algorithm considers a four-dimensional chart of bispans and computes the Viterbi alignment for the minimal valid span (i.e., $[0,0],[1,1]$). Then, it progressively builds larger spans from smaller ones. Let $A_{p,q,u,v}$ represent the Viterbi alignment of the bispan $[p,q],[u,v]$. Each of the four dimensions p,q,u,v of the chart can be manipulated using λ recursions, which guarantees that the new alignment

will be maximized according to IBM Model 1. The following update functions are defined:

- $A_{p,q,u,v+1} = \lambda_{+v}(A_{p,q,u,v})$ inserts one token to the end of the right span with index $v + 1$. Only the most likely alignment for that token is needed, which requires iterating over all the tokens in the left span $[p, q]$. See Figure 3 for an illustration.
- $A_{p,q,u+1,v} = \lambda_{+u}(A_{p,q,u,v})$ removes the first token of the right span with index u. The Viterbi alignment remains the same with the exception that alignments to the removed token at u must be removed. This can be done in time $O(1)$.
- $A_{p,q+1,u,v} = \lambda_{+q}(A_{p,q,u,v})$ inserts one token to the end of the left span with index $q + 1$. All words in the right span must be rechecked, because aligning them to the inserted token at $q + 1$ may yield a better translation score. This update requires $v - u + 1$ operations.
- $A_{p+1,q,u,v} = \lambda_{+p}(A_{p,q,u,v})$ removes the first token of the left span with index p. After removing the token, new alignments must be found for all tokens that were aligned to p. Thus, the number of operations for this update is $K \times (q - p + 1)$, where K is the number of words that were originally aligned to p. In the best case, no words are aligned to the token in p, so nothing needs to be done. However, in the worst case, when all target words were originally aligned to p, this update will result in the recalculation of all alignments.

The algorithm proceeds until all valid cells have been computed, and retrieves the value of the best cell. The most important aspect to consider when updating existing cells is that the update functions differ in complexity, and the sequence of updates

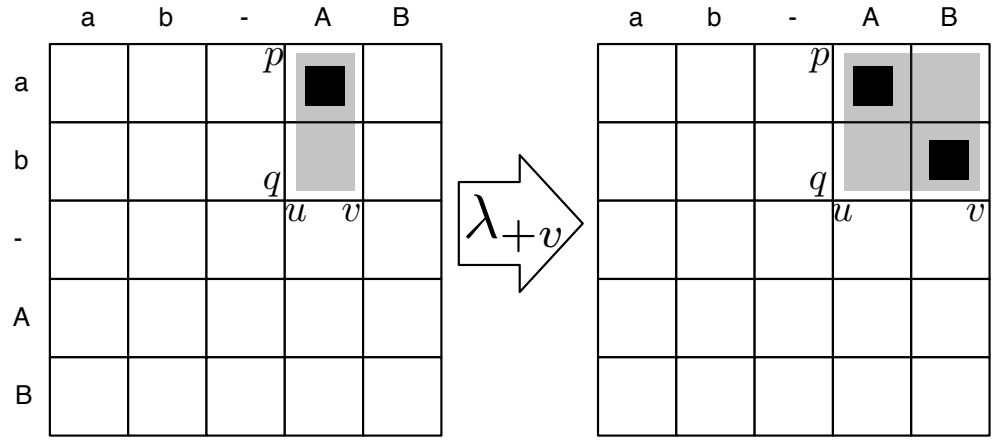

Figure 3
Illustration of the λ_{+v} operator. The light gray boxes show the parallel span and the dark boxes show the span's Viterbi alignment. In this example, the parallel message contains a "translation" of a b to A B.

used defines the performance of the system. Most spans are reachable using any of the four update functions. For instance, the span $A_{2,3,4,5}$ can be reached using $\lambda_{+v}(A_{2,3,4,4})$, $\lambda_{+u}(A_{2,3,3,5})$, $\lambda_{+q}(A_{2,2,4,5})$, or $\lambda_{+p}(A_{1,3,4,5})$. However, it is desirable to apply λ_{+u} whenever possible, because it only requires one operation. Yet, this is not always feasible. For instance, the state $A_{2,2,3,4}$ cannot be reached using λ_{+u}, as the state $A_{2,2,2,4}$ does not exist, since it breaks the condition $p \leq q < u \leq v$. In this situation, incrementally more expensive updates must be considered, such as λ_{+v} or λ_{+q}, which are in the same order of complexity. Finally, we want to minimize the use of λ_{+p}, which may require the recomputation of the Viterbi alignments. Formally, we define the following recursive formulation that guarantees a most likely outcome:

$$A_{p,q,u,v} = \begin{cases} \lambda_{+u}(A_{p,q,u-1,v}) & \text{if } u > q+1 \\ \lambda_{+v}(A_{p,q,u,v-1}) & \text{else if } v > q+1 \\ \lambda_{+p}(A_{p-1,q,u,v}) & \text{else if } q = p+1 \\ \lambda_{+q}(A_{p,q-1,u,v}) & \text{otherwise} \end{cases}$$

This transition function applies the cheapest possible update to reach state $A_{p,q,u,v}$.

If we consider that we always choose the least complex update function, we will reach the conclusion that this algorithm runs at $O(n^4)$ time. Starting by considering the worst update λ_{+q}, we observe that this is only needed in the following cases $[0,1][2,2], [1,2][3,3], \cdots, [n-2,n-1][n,n]$, which amounts to $O(n)$ cases. Because this update is quadratic in the worst case, the complexity of these operations is $O(n^3)$. The update λ_{+p} is applied to the bispans $[*,1][2,2], [*,2][3,3], \cdots, [*,n-1], [n,n]$, where $*$ denotes an arbitrary number within the span constraints but not present in previous updates. Given that this update is linear and we need to iterate through all tokens twice, this update also takes $O(n^3)$ operations in total. The update λ_{+v} is applied for the cells $[*,1][2,*], [*,2][3,*], \cdots, [*,n-1], [n,*]$. Thus, with three degrees of freedom and a linear update, it runs in $O(n^4)$ time. Finally, update λ_{+u} runs in constant time, but is needed for all remaining cases, so it also requires $O(n^4)$ operations. Hence, by summing the executions of all updates, we observe that the order of magnitude of our exact inference process is $O(n^4)$. Note that for exact inference, a lower order would be unfeasible, because simply iterating all possible bispans once requires $O(n^4)$ time.

IBM Model 2 or higher order models were not considered in this problem. Model 2 does require more complexity when computing the most probability alignment for a fixed sentence pair, thus it would increase the complexity of our search algorithm. Recall that λ_{+u} is the preferred update to be used whenever possible during inference, as it only takes one operation. However, this is only true for Model 1. In Model 2, an absolute distortion model must be considered that takes into account the distance between the positions of the right segment and the aligned word in the left segment. However, if we remove the first word in the right segment, we would be shifting the word positions of all words in the right segment, which would require the recomputation of the absolute distortion values of all remaining words in the right segment. This would raise the complexity of this operation to $O(n)$. Thus, with the new set of update functions, the optimal order of updates would require $O(n^5)$ operations.

3.3 Language Pair Filtering

The algorithm just proposed takes the pair of languages as inputs. If we are looking for a particular language pair, this is fine, but when we search for translations in a large

number of language pairs, this is impractical. However, in most cases, many language pairs can be trivially ruled out. For instance, if there are no Han characters, any pair involving Japanese or Chinese can be excluded. Thus, one can efficiently filter out unlikely language pairs without losses.

The IDA model score is composed of the product of the span, language, and translation scores. The computation of the translation score requires checking a lexical translation table in each operation, which is computationally expensive. However, this operation is only expensive if we compute the translation score, as the span and language scores can be more efficiently computed.

Thus, we define the *incomplete* IDA score as:

$$S_{inc}([u,v],r,[p,q],l \mid \mathbf{x}) = \phi_S([p,q],[u,v] \mid l,r,\mathbf{x}) \times \phi_L(l,r \mid [p,q],[u,v],\mathbf{x})$$

We can trivially show that $score([u,v],r,[p,q],l,\mathbf{a} \mid \mathbf{x})$ is bounded by $S_{inc}([u,v],r,[p,q],l \mid \mathbf{x})$, as the incomplete score does not include the product of the translation score, in the $[0,1]$ interval. This means that if we know that the highest *score* value for Chinese–English is 0.4, and wish to check if we can obtain a better score for Arabic–English, we can first compute the S_{inc} score for Arabic–English, and check if it is higher than 0.4. If it is not, we can skip the computation of the translation score for this language pair, as it will not be higher than 0.4. Thus, if we find that the highest S_{inc} score among all possible segmentations of p,q,u,v score for a given language pair l_1,r_1 is lower than *score* for any other language pair, it follows that l_1,r_1 will never be the highest scoring language pair. More formally, we can discard a language pair l_1,r_1 if the following condition applies:

$$\begin{aligned} &\exists_{l \neq l_1, r \neq r_1} \max_{[p,q],[u,v]} S_{inc}([p,q],l_1,[u,v],r_1 \mid \mathbf{x}) \\ &\leq \max_{[p,q],[u,v]} \max_{\mathbf{a}} score([p,q],l,[u,v],r,\mathbf{a} \mid \mathbf{x}) \end{aligned}$$

Our method starts by computing the incomplete IDA scores S_{inc} for all values of r,l. Then, starting from the highest scoring language pairs l,r, we compute their real IDA scores, while keeping track of the highest IDA score. The algorithm can stop once it reaches a language pair whose incomplete IDA score is lower than the highest real IDA score, as we know that they will never achieve the highest real IDA score.

3.4 Evaluation Metrics

The goal of the IDA model is to find the most likely parallel segments $[p,q][u,v]$, their languages l,r, and the alignments $\mathbf{a}$. We also define an evaluation metric that compares the predictions of our model with those obtained by manual annotation. That is, we test whether the predicted values of $[p_h,q_h],l_h,[u_h,v_h],r_h$ correspond to reference values $[p_r,q_r],l_r,[u_r,v_r],r_r$. Because alignments (the $\mathbf{a}$'s) are not of primary interest, we do not evaluate the predictions made by our model.

We start by defining the intersection and union between two segments $[a,b] \cap [a',b']$ and $[a,b] \cup [a',b']$. The intersection between two segments $[a,b] \cap [a',b']$, namely $[a,b]$ and $[a',b']$, computes the number of tokens within the intersection of the intervals, as given by $[\max(a,a'),\min(b,b')]$. Similarly, the union between two segments $([a,b] \cup [a',b'])$ computes the number of tokens within the union of the intervals, given by $[\min(a,a'),\max(b,b')]$. One important aspect to consider is that the segments can span

half words, which can happen if there is a missing space between a sentence pair boundary, such as *uneasyBom*, which contains the English word *uneasy* and the Portuguese word *Bom*. To cope with this, we add the fractional count as the ratio between the number of characters that is included in the interval and total number of characters in the token. Thus, the segment corresponding to *uneasy* in *uneasyBom* would correspond to two-thirds of a single word.

A hypothetical segment $[a_h, b_h]$ with language l_h is scored against the reference $[a_h, b_h]$ with language l_r as:

$$S_{seg}([a_h, b_h], l_h, [a_r, b_r], l_r) = \frac{intersect([a_h, b_h], [a_r, b_r])}{union([a_h, b_h], [a_r, b_r])} \delta(l_h = l_r) \tag{4}$$

This score penalizes the hypothesis segment for each extra token not in the reference, as well as each token in the reference that is not in the hypothesis. Furthermore, segments that differ in language will have a zero score. Unlike our previous work, we decided not to evaluate the language pair detection as a separate task, as only a negligible number of spurious parallel sentences (less than 0.1%) are caused by a incorrect detection of the language pair.

The final score S_{IDA} is computed as the harmonic mean between the segment scores of the parallel segments:

$$S_{IDA}([p_h, q_h], l_h, [u_h, v_h], r_h, [p_r, q_r], l_r, [u_r, v_r], r_r) = \frac{2S_{seg}([p_h, q_h], l_h, [p_r, q_r], l_r) S_{seg}([u_h, v_h], r_h, [u_r, v_r], r_r)}{S_{seg}([p_h, q_h], l_h, [p_r, q_r], l_r) + S_{seg}([u_h, v_h], r_h, [u_r, v_r], r_r)} \tag{5}$$

We opt to compute the harmonic mean (as opposed to a arithmetic mean) because it emphasizes that both parallel segments must be accurate to obtain a high score. This is because parallel segments are only useful when both sides are accurate on the span and language.

4. Parallel Data Extraction

The previous section describes a method for finding the most likely segments p, q, u, v and the language pair l, r and the alignments $\mathbf{a}$, according to the IDA model score, for any document x. However, extracting parallel sentences using this model requires addressing other issues, such as identifying the tweets that contain translations from those that do not. This section will describe how the parallel data extraction process is performed.

We start by assuming that we have access to a sample of tweets (e.g., from Twitter), which we denote as T. The process is divided into three steps. First, we filter the set T in order to remove all monolingual tweets, which will result in a set $T_{mult} \subseteq T$ composed solely of multilingual tweets, which substantially reduces the number of tweets that need to be processed by the following steps. Second, assuming that all tweets in T_{mult} contain parallel data, we extract the parallel segments using the IDA model, and these are placed in the second candidate set $D_{s,t}$. The fact that we are applying the IDA model to tweets that may not contain translations means that many instances in $D_{s,t}$ will not be parallel. Thus, as the last step, we filter these instances, using a classifier that is trained

to predict whether each sample in $D_{s,t}$ is actually parallel. These steps will be described in detail in Sections 4.1, 4.2, and 4.3.

4.1 Filtering

The first step is to filter the set of tweets T so that only multilingual tweets are kept. A tweet is multilingual if it includes more than one language. Thus, tweets passing this filter can contain not only translated material, but also code switching and references to foreign words or names.

Our previous approach for extracting Chinese–English sentence pairs considered only tweets that simultaneously contained a trigram of Chinese characters and a trigram of Latin words. However, this approach is only effective on finding multilingual tweets with languages that do not share the same unicode range. Thus, to allow the extraction of parallel data for language pairs such as English–Spanish, a more general method is needed.

4.1.1 Multilingual Document Detection. Although language identification is largely regarded as a solved problem, traditional language detectors do not identify multiple languages within a document, but instead make the assumption that documents are primarily in one language. Some recent work has begun to address this (Lui, Lau, and Baldwin 2014), and our application imposes an additional efficiency requirement.

We use an approach similar to that used in Section 3.1.3, where we use a word-based language detection approach. The approach is based on estimating the probability that two tokens a and b are in different languages. The probability of a pair of tokens a and b and a pair of respective languages ℓ_a and ℓ_b is given by the product of the probabilities $P_L(\ell_a \mid a) \times P_L(\ell_b \mid b)$, where $\sum_{\ell \in \mathcal{L}} P_L(\ell \mid a) = 1$ and $\sum_{\ell \in \mathcal{L}} P_L(\ell \mid b) = 1$. Thus, the sum of the probabilities of all pairs of language pairs $\sum_{\ell_a \in \mathcal{L}} \sum_{\ell_b \in \mathcal{L}} P_L(\ell_1 \mid a) \times P_L(\ell_2 \mid b)$ is equal to 1, and we can efficiently compute the probability that the pair of tokens are in different languages as:

$$P(\ell_a \neq \ell_b \mid a, b) = 1 - \sum_{\ell \in \mathcal{L}} P_L(\ell \mid a) \times P_L(\ell \mid b) \tag{6}$$

where $P_L(\ell \mid x)$ is once again the probability that token x is in language ℓ, according to a character-based model, and $\mathcal{L}$ is the set of all languages considered. Thus, given a tweet, our model attempts to find a pair of words where $P(\ell_a \neq \ell_b \mid a, b)$ is higher than 0.95. For instance, in the tweet *eu quero ver este cartoon*, where the message is mainly in Portuguese except for the word *cartoon*, which is in English, the model can use the high probability in $P(\ell_{ver} \neq \ell_{cartoon} \mid ver, cartoon)$ to identify this tweet as multilingual. Once again, we are not interested in the accuracy provided by considering contextual information in language detectors for the following reasons. Firstly, the language ambiguity is not a problem, because we are attempting to detect whether a pair is multilingual rather than specifically identifying the languages of the pair of words. Notice that the language of the word *ver* by itself is ambiguous, since both Portuguese and Spanish contain this word. However, the model is only interested in knowing if the language of *ver* is different from the language of *cartoon*. Thus, as long as the probability that the *ver* is low for English, the multilingual language detector will successfully identify this pair as multilingual. Secondly, even if a pair fails to be detected as multilingual, the model will still test all remaining pairs on whether these are parallel.

4.1.2 Word Pair Indexing. Traversing the whole Twitter corpora and computing Equation (6) for all pairs of words can be expensive, given that there may be billions of tweets to process. To cope with this, we use an approach based on the work of Uszkoreit et al. (2010), where we first index T. This index maps a word pair to a list of documents containing that word pair. Next, we traverse the index of word pairs and perform the detection using Equation (7), tagging the list of documents with that word pair as multilingual. Using this approach, we can avoid recomputing the same word pair more than once, if they occur in multiple tweets. Furthermore, we traverse the index so that word pairs that occur in more documents are processed first, and skip word pairs whose associated documents have already been classified as multilingual.

4.2 Location

The set of multilingual tweets T_{mult} is then processed using the IDA model in order to find the most likely values for the variables $[p,q],[u,v],l,r$ that define the parallel segments and their languages. Then, each parallel sentence is extracted and placed in the set $D_{s,t} = \{(s_1,t_1),\ldots,(s_k,t_k)\}$ containing all parallel sentences extracted in the language pair s,t. It is important to note that, whereas the variables l,r denote the languages of the left and right segments, respectively, in the set $D_{s,t}$, we place all parallel sentences that contain the language pair s,t, regardless of their order. Thus, we define the following insertion function:

$$\text{Ins}(\mathbf{x},[p,q],[u,v],l,r,D_{s,t}) = \begin{cases} D_{s,t} \cup \{(\mathbf{x}_p^q, \mathbf{x}_u^v)\} & l = s \wedge r = t \\ D_{s,t} \cup \{(\mathbf{x}_u^v, \mathbf{x}_p^q)\} & l = t \wedge r = s \end{cases} \tag{7}$$

where $\mathbf{x}_a^b$ denotes the segment corresponding to the indexes from a to b in the original tweet $\mathbf{x}$. This function simply checks if the s and t correspond to the left l or right r segments in the detected parallel data, and places the appropriately aligned parallel segments.

Obviously, $D_{s,t}$ will contain a considerable number of non-parallel segments, as multilingual messages in T_{mult} are not guaranteed to contain translated material. Furthermore, we must also consider errors from misalignments of the IDA model and misclassifications of the multilingual message detector. Thus, in order to identify messages that are actually parallel, a final identification step is necessary.

4.3 Identification

Given a candidate sentence pair (s,t), many existing methods for detecting parallel data can be applied (Resnik and Smith 2003; Munteanu and Marcu 2005), as this problem becomes a regular unstructured bitext identification problem. In our initial work (Ling et al. 2013), we simply defined a threshold τ on the IDA model score, which was determined empirically. To obtain better results we train a logistic regression classifier for each language pair, similar to that presented in Munteanu and Marcu (2005), which detects whether two segments are parallel in a given language pair by looking at features of the candidate pair. Training is performed to maximize the classification decisions on annotated candidate pairs.

The classifier uses the following features:

- **IDA model features** - These features correspond to the scores according to each of the factors S_S, S_L, and S_T. Each score is added as a separate feature so that they can be weighted separately.
- **User features** - Although the background of the users that parallel post cannot be determined with absolute certainty, it is safe to assume that they are either bilingual speakers and translate their own messages, or hire translators to translate their posts. Thus, users that do not belong to these categories rarely post parallel messages, since they do not have the means, and likewise, users that are in these categories are likely to post a considerable amount of parallel posts. For instance, Sina Weibo users for Snoop Dogg and Paris Hilton mostly post in English–Chinese. Although the IDA model can align the parallel segments in most cases, some shorter more informal messages, such as *Ready to rock NYC. -* 准备好让纽约嗨起来。, tend to be harder to align and receive a lower score—therefore do not get classified as parallel. These messages tend to be more valuable, however, as they contain artifacts that our translation models cannot translate. Thus, it is desirable to consider the aggregate scores of the user posts as additional information for the classification problem. This is implemented by adding the average IDA model score from all posts from a given user as a feature.
- **Repetition features** - In informal domains, there are many terms that are not translated, such as hashtags (e.g., #*twitter*), at mentions (e.g., @*NYC*), numbers, and people's names. The presence of such repeated terms in the same tweet can be a strong cue for detecting translations. Hence, we define features that trigger if a given word type occurs in a pair within a tweet. The word types considered are hashtags, at mentions, numbers, and words beginning with capital letters.
- **Length feature** - It has been known that the length differences between parallel sentences can be modeled by a normal distribution (Gale and Church 1991). Thus, we used parallel training data (used to train the alignment model) in the respective language pair to determine $(\tilde{\mu}, \tilde{\sigma}^2)$, which lets us calculate the likelihood of two hypothesized segments being parallel.

For each language pair s, t, we train separate classifiers for each language pair on annotated parallel data $D_{gold}(s, t)$. The method used to obtain the necessary annotations is described in Section 5.

Intrinsic evaluation. The quality of the classifier can be determined in terms of precision and recall. We count one as a true positive (*tp*) if we correctly identify a parallel tweet, and as a false positive (*fp*) if we spuriously detect a parallel tweet. Finally, a true negative (*tn*) occurs when we correctly detect a non-parallel tweet, and a false negative (*fn*) if we miss a parallel tweet. Then, we set precision as $\frac{tp}{tp+fp}$ and recall as $\frac{tp}{tp+fn}$. Afterwards, F-measure is used to test the overall accuracy of the system in terms of precision and recall.

Depending on the nature of the task to which the extracted corpora is applied, recall may be more important than precision or vice-versa. For instance, as training data for MT models, recall tends to matter more, as phrase-based MT models are robust to errors in the training data.

5. Crowdsourcing Parallel Segment Identification

Section 4.3 presented a supervised method to train a classifier that discriminates parallel and non-parallel data. However, training requires annotated instances where parallel and non-parallel segments are identified. Furthermore, to evaluate the quality of both the location and identification tasks, we also need annotated data. The same can be said about evaluating the quality of MT, which typically uses a gold standard of translated corpora.

Unfortunately, it is difficult to find such material for all language pairs, especially in the microblog domain. Using expert annotators is expensive and it may be hard to find bilingual speakers for many language pairs. We will rely on crowdsourcing to produce annotated data sets, which has been successfully used to generate parallel data (Ambati and Vogel 2010; Zaidan and Callison-Burch 2011; Post, Callison-Burch, and Osborne 2012; Ambati, Vogel, and Carbonell 2012). However, these methods have been focused on using workers to translate documents. Our classifier is ideally trained with tweets labeled with whether they contain parallel data.

5.1 Labeling Parallel Tweets

To define a set of tweets to annotate for each language pair, we use the IDA model to find the highest scoring language pair of that tweet (assuming it contains translated material). This allows us to set up tasks for specific language pairs, so that only workers who understand both languages will take them.

The selected tweets are then placed in sets of 21 questions, which ask the worker whether a tweet contains translated material in the given language pair. Additionally, another four reference questions (whose answers are known) are added to the set to evaluate the worker's quality. Jobs are accepted if the worker can answer at least three of the four reference questions correctly. Finally, each task is performed until five workers are accepted and the final result for each tweet is given by a weighted average of the answers of all five workers. More formally, we compute the weighted average given by $\frac{\sum_{i=1..N} \delta_p(i)w(i)}{\sum_{i=1..n} w(i)}$, where δ_p is 1 if answer i is positive and 0 otherwise, and $w(i)$ is the weight of the worker. The weight $w(i)$ is defined as the ratio of correct answers from job i in the reference set R, given by $\frac{c}{R}$. Finally, if the weighted ratio is higher than 0.5, we label the tweet as being parallel, otherwise it is labeled as negative.

5.2 Obtaining High-Quality Bitext

In the previous task, we obtain judgments as to whether a particular tweet contains parallel data or not. In order to assess the quality of the identified spans, we add an additional task, where workers retrieve the bitext within the tweets that were previously identified to contain such data.

Tweets containing parallel text are placed in sets of 21 questions with 4 reference questions, and each question asks the worker to identify the indices of the parallel segments. Once again, workers are accepted based on their performance on the four

reference questions. More specifically, we use the evaluation metric defined in Section 3.4 to measure how the character indexes defined by the worker perform against the reference annotation. To set a minimum score for each reference, we degrade the reference so that the smallest non-parallel segment is placed within the closest parallel segment. For instance, in the Korean–English example in Figure 1, the segment *Hahah* would be merged with the English parallel segment. If the worker's scores are higher than the pseudo reference's scores with respect to the reference, the job is accepted. Finally, each task is performed until two workers are accepted, and the annotations of the worker that scored highest against the reference are chosen.

6. Sina Weibo Data Crawling

In the previous sections, we described a method to obtain parallel data from microblog posts. Yet, obtaining a large data set from such environments is challenging in itself because of rate limits imposed on most users. In this section, we will describe the crawling method we used to crawl 3M parallel Chinese–English sentence pairs from Sina Weibo within 6 months.

The main problem we have to address relates to the fact that a random sample of over 1.6 billion tweets from Twitter only yields approximately 300K English–Chinese sentence pairs. However, because of the rate limiting[3] established by Sina Weibo's API, we are only able to send 150 requests per hour. Each request can fetch up to 100 posts from a user, and subsequent sets of 100 posts request additional API calls. This means that crawling 1.6 billion tweets is beyond our capabilities. In fact, we were only able to crawl approximately 90 million tweets in 6 months.

Thus, we wish to optimize the parallel tweets we obtain from each API call. We do this by observing that some users post many messages in parallel, whereas others post none. Thus, we use one request to obtain the most recent 100 messages a user has posted. We run the IDA model on that sample of messages to determine if they contain any parallel data.[4] If the number of automatically identified parallel messages within those 100 tweets is higher than 10, that user becomes a crawl target. We obtain all messages from crawl targets, and we periodically check if new messages have been posted.

The crawler operates as follows:

1. Pick a random user and crawl 100 posts.
2. For all crawl targets that have not been updated within the last week, check and crawl their new posts.
3. Repeat from 1.

During the one-hour downtime from exhausting the 150 requests in steps 1 and 2, we run the following operations:

1. Run the IDA model for users with unprocessed tweets.
2. Set the user as a crawl target if more than 10% of their tweets are parallel.

3 `http://open.weibo.com/wiki/API文档/en`.

4 The identification of parallel messages was performed by setting a threshold on the IDA score rather than the full classification-based approach. It is not possible to repeat this method under the same conditions because of the dynamic nature of microblogs.

This is done separately in order to spend the 150 requests when available, as unspent requests are lost after an hour.

7. Experiments

In this section, we describe the experiments performed to show the effectiveness of our proposed algorithm and the value of the data obtained. There are three sets of experiments that are performed. We evaluate the parallel data extraction process intrinsically by testing each of the three steps described in Section 4, and extrinsically by testing its utility when used as training data for MT systems.

7.1 Set-up

We consider the following languages: Arabic, German, English, French, Japanese, Korean, Chinese, Portuguese, Russian, and Spanish. On Sina Weibo, we focus on extracting sentence pairs involving Chinese as one of the elements in the pair, as Chinese is the main language in Sina Weibo. On Twitter, we focus on finding sentence pairs involving English as one of the languages.

7.1.1 Tokenization. Tokenization converts a tweet into a sequence of tokens. Our tokenizer must consider a variety of languages and some common artifacts in the microblog domain. General properties of the tokenizer we used include:

- Sequences of Latin, Arabic, and Cyrillic characters are separated into tokens using white spaces.
- Each Han, Hangul, and Kana character is considered an independent token.
- Numbers are considered a token. Quantifiers, such as *$* and *kg*, are separated in a different token.
- Http links, hashtags, and emoticons are standardized into _HTTP_, _HASH_, and _EMO_ tokens.
- Punctuation marks are considered separate tokens.
- Traditional and Simplified Chinese characters are standardized into Simplified Chinese characters.[5]

One particular aspect in our tokenizer is that it stores the starting and ending offsets of the tweet from which each token was extracted. This is done so that after finding the parallel segments relative to the tokens, we can also use these offsets to recover the parallel segments in the non-tokenized tweet.

7.1.2 Language Detection. A character-based language detector is required for the calculation of the language score in Section 3.1.3 and for the multilingual tweet detection in Section 4.1. This detector is trained on 112 languages, with the monolingual data extracted from Wikipedia. Although we do not extract parallel sentences for all the

5 In general, Traditional and Simplified characters convey the simple meaning and normalizing them improves alignments by reducing sparsity.

112 languages, more information regarding existing languages allows the detector to estimate the language probabilities more accurately. As we are using a character trigram model, a large amount of data is not required to saturate the model probabilities. Thus, for each language, we extract all data from Wikipedia up to a limit of 100K lines in order to keep the model compact.

7.1.3 Translation Lexicons. The IDA model uses translation lexicons to determine the translation score, as described in Section 3.1.1, which are estimated using parallel corpora. More specifically, we use the aligner described in Dyer, Chahuneau, and Smith (2013) to obtain the bidirectional alignments from the parallel sentences. Afterwards, we intersect the bidirectional alignments to obtain sure alignment points. Finally, we prune the lexicon using significance pruning (Johnson et al. 2007) with the threshold $\alpha + \epsilon$ (as defined in that work). The intersection and the pruning are performed to reduce the size of the lexicon to make the look-up faster. The breakdown of the different lexicons built is shown in Table 1.

7.2 Building Gold Standards

To train and test the classifier described in Section 4.3, and perform MT experiments, a corpus of annotated tweets is needed for different language pairs. Table 2 summarizes the annotated corpora for the two domains (column *Source*) and the different language pairs (column *Language Pair*). We also report the method used to obtain the annotations (column *Method*), where *Expert* denotes the manual annotation from native speakers, and exampCrowdsourcing denotes data that were crowdsourced using Amazon Mechanical Turk. The number of tweets that were annotated and the number of parallel sentences that were found are reported in columns #*Annotatated Tweets* and #*Parallel*, respectively. Finally, we also illustrate the average size of the English and Foreign sentences in the parallel data after tokenization in columns *Average Size (English)* and *Average Size (Foreign)*. We observe that the number of words in Twitter data sets are smaller than those in Weibo, which can be explained by the fact that posts in Twitter are limited to 140 characters, whereas in Sina Weibo, this restriction is not enforced.

It is important to keep in mind that these numbers are not fully representative of the Twitter data as a whole, as we are filtering monolingual tweets prior to annotation. Thus, we cannot draw conclusions about the ratio between parallel and non-parallel

Table 1
Lexicons built using parallel corpora.

Language Pair	Corpus	# sentence pairs
German–English	EUROPARL	1,920K
Spanish–English	EUROPARL	1,966K
French–English	EUROPARL	2,007K
Portuguese–English	EUROPARL	1,687K
Chinese–English	FBIS	300K
Arabic–English	NIST	970K
Russian–English	Yandex	1,000K
Korean–English	KAIST	60K
Korean–Chinese	KAIST	60K
Japanese–English	Tatoeba	150K

Table 2
Description of the annotated data.

Source	Language Pair	Method	#Annotated Tweets	#Parallel Sentences	Average Size (English)	Average Size (Foreign)
Weibo	English–Chinese	Expert	4347	2581	18.09	28.45
Twitter	English–Chinese	Crowdsourcing	2688	1302	8.76	14.03
Twitter	English–Arabic	Crowdsourcing	2520	1758	8.32	6.84
Twitter	English–Russian	Crowdsourcing	5082	1631	7.11	6.19
Twitter	English–Korean	Crowdsourcing	3906	1116	7.10	15.67
Twitter	English–Japanese	Crowdsourcing	2583	1155	10.22	19.07
Twitter	English–Portuguese	Crowdsourcing	2583	542	8.50	9.02
Twitter	English–Spanish	Crowdsourcing	2541	722	7.09	7.28
Twitter	English–French	Crowdsourcing	2856	594	8.46	8.95
Twitter	English–German	Crowdsourcing	2772	909	8.94	7.31

data in Twitter. For instance, the ratio between parallel and non-parallel tweets for Arabic–English in Twitter is 2:1 in the annotated data sets, which is definitely not the case in a uniformly extracted data set.[6] However, performing the annotation in a uniformly extracted data set is problematic for two reasons. Firstly, a huge annotation effort would be required to find a significant number of tweets containing translations, since this is only the case for a minority of tweets. Secondly, training the classifier on such an unbalanced data set would bias the classifier towards the negative case, as the majority of the samples belong in this category, which is not desired.

7.3 Parallel Data Extraction Experiments

We report on the experiments performed in each of the stages of the parallel data extraction process described in Section 4.

7.3.1 Filtering. The filtering step (described in Section 4.1) attempts to filter out monolingual tweets, because these are sure to not contain parallel data. Ideally, we would uniformly sample tweets and annotate them on whether they are multilingual. However, this would require an extraordinary amount of effort to obtain a sample with a large number of multilingual tweets, as most tweets are monolingual. Thus, we use a pre-existing annotated data set from Twitter, where each word in the tweet was annotated with its language. In this data set, there are 773 annotated samples from Twitter. We filter these so that all tweets contain words in at least two different languages, resulting in 554 multilingual tweets. From this data set, we define two splits. The first one contains only the language pairs that we are extracting parallel data from in this work, which allows us to estimate the degree our algorithm is spuriously removing multilingual tweets in the filtering step. In all, 291 tweets from the 554 were obtained according to this criteria. The second subset is restricted to languages that use the Latin alphabet[7] as these are more difficult to label correctly. This subset contains 222 tweets. Finally, to build a data set of monolingual tweets, we sample tweets from Twitter uniformly until we find 2,000 tweets that are monolingual.

6 We obtain this ratio as we are filtering monolingual tweets, and we are removing samples that do not contain alignment points.
7 English, Spanish, French, German, and Czech.

These are then injected into the Twitter data set, and we compute the percentage of tweets that were labelled correctly from the different sets, using different thresholds for Equation (7).

Results are shown in Figure 4, where the plot line *Multilingual* represents the percentage of multilingual tweets from the full set of 554 tweets that were correctly labelled (y-axis) for different values for different thresholds (x-axis). Plot lines *Multilingual (Used Languages)* and *Multilingual (Latin)* denote that percentage of correctly labelled multilingual tweets from the subsets created by restricting the 554 tweets to the used languages in this work for parallel data extraction and the set consisting of Latin languages, respectively. Finally, the plot line *Monolingual* represents the set of monolingual tweets that were correctly labelled. We can observe that by simply removing the top 90% of word pairs, we can remove 67.8% of the monolingual tweets at the cost of losing 10–15% of the multilingual tweets at threshold 0. When we start increasing the threshold, we observe a substancial improvement for the detection of monolingual tweets, at the cost of mislabelling multilingual tweets. As expected,

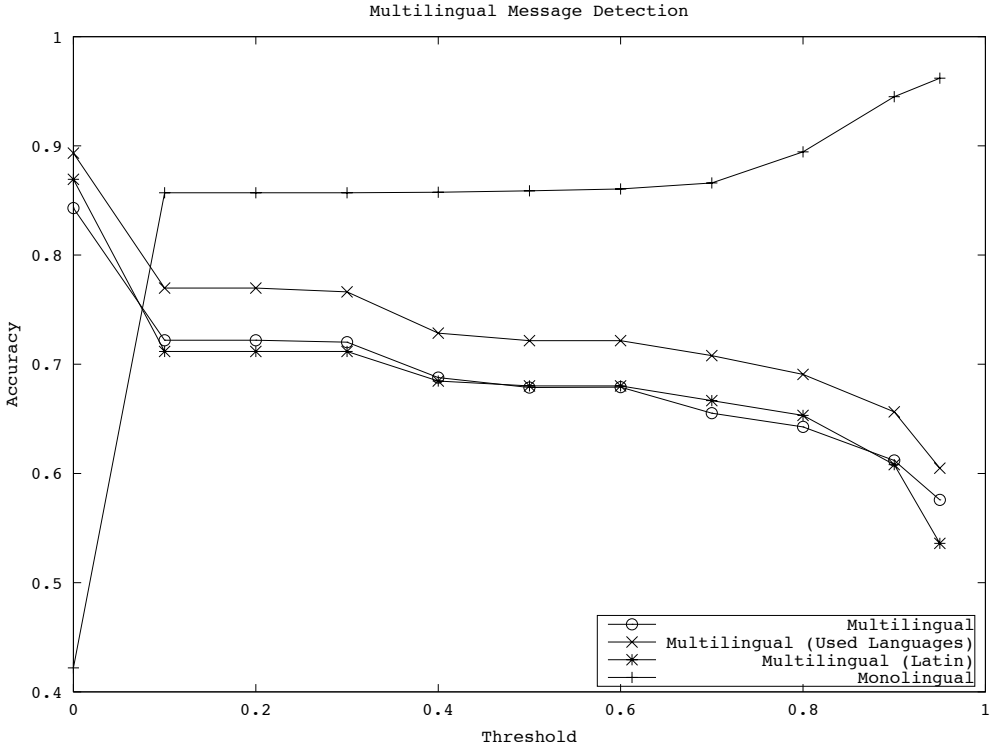

Figure 4
Results for the language based filtering task. The x-axis represents threshold for Equation (7). The Multilingual line represents the percentage of multilingual tweets that are kept, and the Monolingual line represents the percentage of monolingual tweets that are discarded. Thus, we wish to maximize the Monolingual score and minimize the Multilingual score. For contrast, we also show the same scores using the languages that we are extracting parallel data for in the Multilingual (Used Languages) line, and those that are in Latin languages in the Multilingual (Latin) line.

the task is slightly hard for Latin languages, because of the orthographic similarities between these languages.

In our work, we set the threshold to 95% in order to maximize the number of monolingual tweets that are removed. Although this produces worse results for the detection of multilingual tweets, it substantially reduces the number of tweets that need to be processed. Furthermore, a large portion of the misclassifications for multilingual tweets are the result of tweets that only contain one or two words in a different language, such as a person's name, and these tweets are not likely to contain parallel data.

7.3.2 Location Results. To test the quality of the location of the parallel segments, we compare the automatic segment boundaries with the human annotations using the S_{IDA} metric defined in Section 3.4, which measures the overlap between the proposed and the reference boundaries. Results for different language pairs and domains can be found in Table 3, where the average overlap score S_{seg} for the English and Foreign segments are shown in columns *English Overlap* and *Foreign Overlap*, respectively. The S_{IDA} score obtained as a harmonic mean between the previous scores is shown in the S_{IDA} column.

From these results we can see that for Sina Weibo's English–Chinese corpus, the results are significantly higher than those in Twitter. One explanation for this is the fact that parallel segments found in Weibo are longer. This allows the alignment model to find more word alignments, which can be used to find better boundaries for the parallel spans.

We can also observe that results tend to be higher on the language pairs, where more parallel data were used to train the lexical translation table. For instance, in the English–Korean and English–Japanese language pairs, where the parallel corpora used consisted of only 60K and 150K sentence pairs, the results are evidently worse compared to the results obtained for the English–Arabic and English–Russian language pairs, where approximately 1 million sentence pairs were used.

7.3.3 Identification Results. The aim of the identification task is to determine whether a given tweet contains parallel data. We used the data sets described in Section 7.2, and these were evenly divided into training and test sets. The max-entropy classifier was

Table 3
Results for the location of the parallel segments over different data sets. The *English Overlap* and *Foreign Overlap* columns illustrate the average of the overlaps of the automatically extracted segments for the English and Foreign segments, respectively. The final score is computed as the harmonic mean between the two previous overlaps, which is shown in the S_{IDA} column.

Source	Language Pair	English Overlap	Foreign Overlap	S_{IDA}
Weibo	English–Chinese	0.848	0.891	0.859
Twitter	English–Chinese	0.743	0.779	0.760
Twitter	English–Arabic	0.801	0.794	0.771
Twitter	English–Russian	0.793	0.818	0.778
Twitter	English–Korean	0.737	0.744	0.706
Twitter	English–Japanese	0.695	0.786	0.704
Twitter	English–Portuguese	0.759	0.781	0.770
Twitter	English–Spanish	0.798	0.795	0.796
Twitter	English–French	0.836	0.809	0.822
Twitter	English–German	0.734	0.718	0.726

trained using Weka (Hall et al. 2009), which maximizes the weighted F-measure of the positive and negative samples. We calculate the precision, recall (on positive labels), and accuracy at increasing intervals of 10% of the top scoring samples.

Results for the English–Chinese language pair for both the Twitter and Sina Weibo domains using the full set of features are presented in Figure 5. We can observe that Weibo contains a larger number of parallel tweets as the precision when all the data is considered parallel is only 46% for Twitter, compared with 61% in Weibo. This is because the Twitter data set we used to extract the parallel data from was extracted uniformly, whereas the Sina Weibo tweets were crawled using the algorithm described in Section 6, which attempts to maximize the amount of tweets containing translations while crawling. We can also observe that results are significantly better for Weibo, as the precision curve for the Weibo data set is consistently higher than that for the Twitter data set.

It is not the goal of this work to exhaustively engineer features to maximize the results for this task, but we wish to show that additional features can be used to improve the quality of the classifier, some even containing Twitter or Weibo specific attributes, such as the meta-information regarding the user that posted each tweet. To do this, we trained the max-entropy classifier using an increasingly larger set of features and present the weighted average of the F-measure for positive and negative labels.

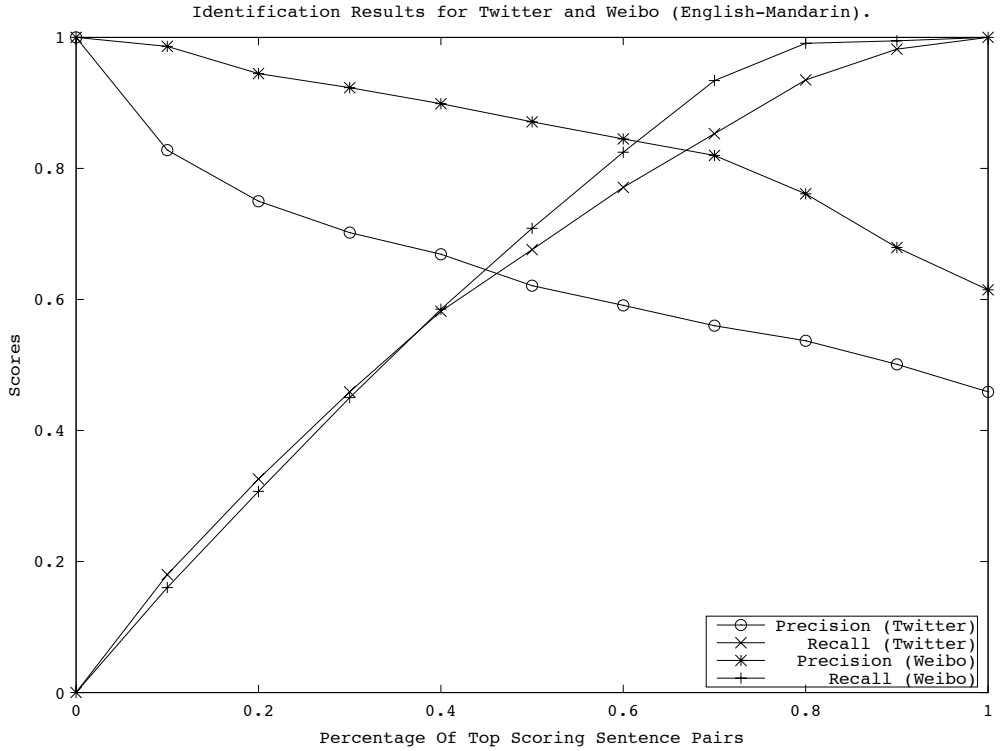

Figure 5
Precision and recall curves for an increasingly larger number of top scoring sentence pairs for the Weibo and Twitter data sets for the English–Chinese language pair. The precision and recall scores are quantified on the *y*-axis, and the percentage of samples labeled as positive is represented on the *x*-axis.

Table 4
Results for the parallel data identification task over different data sets. The columns present the identification results using an incremental set of features. Each cell contains the F-measure using a given data set and set of features.

Source	Language Pair	IDA	+User	+Length	+Repetition
Weibo	English–Chinese	0.781	0.814	0.839	**0.849**
Twitter	English–Chinese	0.599	0.598	0.603	**0.652**
Twitter	English–Arabic	0.721	0.721	0.725	**0.763**
Twitter	English–Russian	0.692	0.705	0.706	**0.729**
Twitter	English–Korean	0.635	0.650	0.650	**0.655**
Twitter	English–Japanese	0.570	0.566	0.569	**0.579**
Twitter	English–Portuguese	0.859	**0.860**	0.857	0.858
Twitter	English–Spanish	0.841	0.849	0.850	**0.850**
Twitter	English–French	0.886	0.886	0.885	**0.888**
Twitter	English–German	0.789	0.787	0.794	**0.798**

Results for different language pairs are shown in Table 4, where we can observe that results can be improved by adding richer features, similar to previous work (Resnik and Smith 2003; Munteanu and Marcu 2005). The microblog-specific *User features* seems to work best in Sina Weibo, as the crawling methodology we used would obtain a large number of posts from the same user. On the other hand, the Twitter data set was obtained from a uniform crawl where fewer tweets from the same user can be found, which is why less significant improvements were observed. We can also observe incremental improvements from widely used features for this task (Length, Repetition, and Language).

The quality of the results of the identification task is strongly correlated to the quality of the location task, as the identification task depends on the quality of the automatically detected parallel segments. For instance, we can observe better results for Arabic–English, which also obtained a high S_{IDA} in Table 3. Furthermore, similar language pairs, such as English–Spanish and English–French, tend to be better aligned, and therefore obtain higher scores. In these cases, additional features are less beneficial for the classifier.

7.3.4 Data Representation. To provide an interpretable view of the contents of the parallel data that was crawled, we look at the distribution over topics of the parallel data set inferred using latent Dirichlet allocation (LDA) (Blei, Ng, and Jordan 2003). Thus, we grouped the Weibo filtered tweets by user, and ran LDA over the predicted English segments, using 12 topics. The seven most interpretable topics are shown in Table 5. We see that the data contain a variety of topics, both formal (Chinese news, religion) and informal (entertainment, music).

7.4 Machine Translation Experiments

In order to measure the impact of the extracted corpora, we perform an extrinsic experiment where we use the extracted data as training parallel sentences for existing MT systems. We first perform an extensive test on the English–Chinese language pair, where we show that the extracted corpus contributes to improve the state-of-the-art

Table 5
Most probable words inferred using LDA in several topics from the parallel data extracted from Weibo. Topic labels (in parentheses) were created manually for illustration purposes.

Topic	Most probable words in topic
1 (Dating)	love time girl live mv back word night rt wanna
2 (Entertainment)	news video follow pong image text great day today fans
3 (Music)	cr day tour cn url amazon music full concert alive
4 (Religion)	man god good love life heart would give make lord
5 (Nightlife)	cn url beijing shanqi party adj club dj beijiner vt
6 (Chinese News)	china chinese year people world beijing years passion country government
7 (Fashion)	street fashion fall style photo men model vogue spring magazine

results in Twitter, Sina Weibo, and also more formal domains. Then, we show that these results generalize for other language pairs.

To accurately quantify the effect of our corpora in translation, all experiments will be conducted using fixed development and test sets for each domain as well as the same system set-up but with different training sets.

7.4.1 Data Sets. We report on machine translation experiments using our harvested data in three domains: edited news, Twitter, and Sina Weibo.

- **News translation** - For the news test, we created a new test set from a crawl of the Chinese–English documents on the Project Syndicate Web site,[8] which contains news commentary articles. We chose to use this data set, rather than the more standard NIST test sets, to ensure that we had recent documents in the test set (the most recent NIST test sets contain documents published in 2007, well before our microblog data were created). We extracted 1,386 parallel sentences for tuning and another 1,386 sentences for testing, from the manually alignment segments. For this test set, we used 8 million sentences from the full NIST parallel data set as the language model training data. We call this test set **Syndicate**.
- **Sina Weibo translation** - The Sina Weibo corpus was created by using the gold standard annotations described in Section 7.2. This contained 2,581 parallel sentences, where 1,290 pairs were used for tuning and the other 1,291 pairs were used for testing. Naturally, we removed all these instances from the training data. We refer to this test set as **Weibo**.[9] The language model used in this work was built using 10 million tweets from Sina Weibo for Chinese. As for English, we used 10 million tweets from Twitter.
- **Twitter translation** - The Twitter data sets were built using a similar methodology, where we use the gold standard annotations to create a held-out data set of parallel sentences. In all cases, we split the held-out

8 `http://www.project-syndicate.org/`.
9 We acknowledge that self-translated messages are probably not a typically representative sample of all microblog messages. However, we do not have the resources to produce a carefully curated test set with a more broadly representative distribution. Still, we believe these results are informative as long as this is kept in mind.

data set evenly to obtain the tuning and testing data sets. To build the English and Chinese language models, we used the same data as the Sina Weibo data set.

7.4.2 Training Data. We report results on these test sets using different training data. First, we use the FBIS data set, which contains 300K high quality sentence pairs, mostly in the broadcast news domain. Second, we use the full 2012 NIST Chinese–English data set (approximately 8M sentence pairs, including FBIS). Finally, we use our crawled data from Sina Weibo (referred to as Weibo) and those extracted from Twitter (referred to as Twitter) by themselves but also combined with the two previous training sets. The max-entropy classifier for detecting parallel data was tuned for 50% precision, where 113K Twitter parallel sentences from Twitter and 800K sentence pairs from Sina Weibo were extracted.

7.4.3 Set-up. We use the Moses phrase-based MT system with standard features (Koehn, Och, and Marcu 2003). As the language model, we use a 5-gram model with Kneser-Ney smoothing. The weights were tuned using MERT (Och 2003). Results are presented with BLEU-4 (Papineni et al. 2002).

7.4.4 Results. The BLEU scores for the different parallel corpora are shown in Table 6 and the top 10 out-of-vocabulary words for each data set are shown in Table 7. Results for the microblog test sets (Weibo and Twitter) suggest that considerable improvements can be obtained by using the crawled corpora for this domain.

The most notable result can be observed by contrasting the obtained BLEU scores on the Weibo test set with the NIST and Weibo training corpora, where improvements ranging from 250% to 300% can be observed. Although this is a promising result, it is important to consider the fact that we are drawing training and test samples from the same domain, which naturally leads to better results than using training corpora that were drawn from other domains. In some cases, these improvements can also be the result of overfitting. However, a strong indication that this is not the case is the fact that similar results can be obtained for the Twitter test set, where we can observe a BLEU improvement from 9.55, using the NIST corpus, to 23.57, using the Weibo corpus. This shows that the extracted corpus contains many translated elements that are representative of the microblogging domain that are not found in publicly

Table 6
BLEU scores for different data sets in different translation directions (left to right), broken out with different training corpora (top to bottom).

	Syndicate		**Weibo**		**Twitter**	
	ZH-EN	EN-ZH	ZH-EN	EN-ZH	ZH-EN	EN-ZH
FBIS (300K)	8.86	19.69	9.24	13.60	8.44	13.04
NIST (8M)	**10.87**	**22.32**	10.76	14.89	9.55	13.97
Twitter (113K)	2.90	6.59	10.53	11.42	12.67	15.43
Weibo (800K)	9.78	20.72	**33.32**	**34.48**	**23.57**	**25.01**
NIST+Twitter	12.43	23.81	15.29	16.72	15.83	19.29
NIST+Weibo	**13.11**	**24.60**	**33.01**	**34.33**	**23.35**	**27.07**

Table 7
The most frequent out-of-vocabulary words and their counts in the test set for the three English-source test sets (Syndicate, Weibo, and Twitter) with four different training sets (FBIS, NIST, Weibo, and Twitter). We did not consider http links, hash tags, and at mentions in this list as these are generally not translated.

Syndicate (test)			
FBIS	NIST	Weibo	Twitter
obama (83)	barack (59)	democracies (15)	cambridge (80)
barack (59)	namo (6)	imbalances (13)	debt (68)
princeton (40)	mitt (6)	mahmoud (12)	2008 (55)
ecb (8)	guant (6)	millennium (9)	monetary (52)
bernanke (8)	fairtrade (6)	regimes (8)	economies (47)
romney (7)	hollande (5)	wolfowitz (7)	paris (45)
gaddafi (7)	wikileaks (4)	revolutions (7)	increasingly (41)
merkel (7)	wilders (3)	qaddafi (7)	princeton (40)
fats (7)	rant (3)	geopolitical (7)	recession (39)
dialogue (7)	esm (3)	genome (7)	fiscal (38)
Weibo (test)			
FBIS	NIST	Weibo	Twitter
2012 (24)	showstudio (9)	submissions (4)	alanis(15)
alanis (15)	crue (9)	ivillage (4)	mexico (14)
crue (9)	overexposed (8)	scola (3)	ego (12)
showstudio (9)	tweetmeian (5)	rbst (3)	kelly (11)
overexposed (8)	tvd (5)	curitiba (3)	showstudio (10)
itunes (8)	iheartradio (5)	zeman (2)	awakening (9)
havoc (8)	xoxo (4)	yaptv (2)	milan (8)
sammy (6)	snoop (4)	witnessing (2)	crue (8)
obama (6)	shinoda (4)	whoohooo (2)	wit (7)
lol (6)	scrapbook (4)	wbr (2)	trespassing (7)
Twitter (test)			
FBIS	NIST	Weibo	Twitter
twitter (5)	twitter (5)	zzzzzzzzz (1)	submit (3)
kyuhyun (5)	kyuhyun (5)	unobstructive (1)	fin (3)
siwon (4)	siwon (4)	totalitarian (1)	behavior (3)
lol (4)	oppa (3)	telus (1)	artificial (3)
oppa (3)	didn (3)	takamine (1)	scribd (2)
haha (3)	scribd (2)	spansh (1)	owes (2)
gd (3)	omg (2)	spaciousness (1)	net (2)
didn (3)	kpop (2)	soshi (1)	kindness (2)
wanna (2)	facebook (2)	snowkyu (1)	february (2)
tid (2)	exo (2)	wukan (1)	ego(2)

available corpora, such as NIST. Examples include translations for newer terms, such as *iTunes*. Table 7 illustrates the list of terms in each of the test sets (*Syndicate*, *Weibo*, and *Twitter*) that could not be found in the respective training data (*FBIS*, *NIST*, *Weibo*, and *Twitter*). Although there is no guarantee that all the terms are translated correctly, it is a promising result that, using the Weibo corpus, we can find translations for all words that occur more than once in the Twitter data set. We can observe that

the NIST and FBIS data sets do not possess frequent terms, such as *wanna*, *lol*, and *omg*, and also newer terms like *kpop*[10] in their translation inventory. An aspect that does not reflect in Table 7 are words with the same orthography, such as *u* and *4*, which are found in the NIST data set, but can be used with different meanings in microblogs. We also found examples of slang terms that are not addressed by the NIST data set, and are learned from the Weibo training corpus. First, we observe the character *jiǒng*, which generally means *embarrassed* in informal speech. The term 粉丝 (*fěn sī*) is a phonetic translation of the English term *fans*. This is translated incorrectly using the FBIS data set as *powder silk*, which is its literal character-level translation. Finally, the term *diǎo sī* is very common in Chinese microblogs. There is no direct translation for this term, but it is generally translated based on context into *loser* or *sucker* in our extracted parallel sentence pairs from Weibo.

If we observe the results using the Twitter training set (Twitter row), we can observe that improvements are much more reduced. In fact, for the Weibo test set, the obtained results are lower than using the NIST data set. This can be explained by the lexical gaps caused by the small size of the Twitter training set (117K sentence pairs). However, we can see that a considerable improvement can be obtained by combining these data sets (NIST+Twitter row), which suggests that a meaningful translation inventory can be obtained from this training set for the translation of microblog data.

As for the Syndicate test set (Syndicate row), the NIST and FBIS data sets perform better than our extracted parallel data, as they are much more representative of this domain. Yet, we can observe that training with the Weibo corpus still obtains a similar result compared to the NIST and FBIS data sets, due to the fact that many news events are posted on Weibo and thus contain the formal language present in NIST and FBIS. Furthermore, combining data sets leads to improvements in BLEU. Error analysis indicates that one major factor is that names from current events, such as *Romney* and *Wikileaks*, do not occur in the older NIST and FBIS data sets, but are represented in the Weibo data set.

7.4.5 Experiments for Other Language Pairs. To further validate the generalizability of our algorithm, we perform the same test on the other test data we obtained from Twitter, as the Weibo data set mainly included English–Chinese sentence pairs (because the crawler was parameterized to find users who post in this language pair). The Twitter data set was crawled uniformly, which lowered the amount of training data that can be found within this data set. We do not perform experiments for language pairs where the number of extracted training sentences is particularly small (3K for Korean–English). For other language pairs, we test whether we can improve translation results by adding the extracted corpora into formal data sets.

Once again, the crowdsourced corpus in Table 2 is divided evenly into development and test sets for MT. The in-domain training corpus for Twitter is extracted automatically by optimizing the threshold for 50% precision. The size of these corpora varies from 10K to 150K. As out-of-domain training data, we use the data from which the lexicons (Table 1) were built. As monolingual data for English, we use the same 10M tweets from Twitter as in the previous experiment, and we always translate from the foreign language into English.

Results are shown in Table 8, where we can observe that, on average, the use of the in-domain data results in better translation quality, as expected. Furthermore,

10 Korean pop music.

Table 8
Translation experiment results for different language pairs on the Twitter data. Each column shows the MT results for a given source language into English. The *Out-of-domain size* and *In-domain size* rows represent the number of sentence pairs in the in-domain and out-of-domain training sets. The *Out-of-domain* and *+In-domain* rows show the BLEU scores for a set-up where only the out-of-domain data was used and the same set-up after adding the in-domain data set, respectively.

Source Language	Arabic	Spanish	French	Japanese	Russian
Out-of-domain size	970K	1966K	2007K	150K	1000K
In-domain size	138K	36K	12K	31K	11K
Out-of-domain	12.76	11.11	16.01	6.72	28.05
+In-domain	29.27	27.00	25.89	11.23	31.53

combining both the in-domain and out-of-domain data improves the results further, similarly to the Chinese–English experiments.

In most cases, the significant improvements are led by the non-existence of frequently used variations for fundamental function words in each language, which do not occur in formal domains. For instance, in the Portuguese–English language pair, the Portuguese terms *muito* and *para*, which mean *very* and *for* in English, are generally written as *mto* and *pra* in Twitter. This can be seen on the Portuguese–English example in Figure 1. Because the in-domain corpus that is used is relatively small, adding a large amount of out-of-domain bitext yields large improvements in BLEU.

7.4.6 Translation of Online Terms. In order to provide insight of the distintive translations of online terms that are learned in our extracted data, we manually selected some examples of parallel sentences containing terms that are generally not found in other media. These are shown in Figure 6.

The first row provides a sentence pair with the term *diaosi*, which is a popular buzzword among Chinese communities and is used to describe a class of underprivileged men, lacking certain desirable features (looks, social status, wealth, etc.). Generally, these terms are difficult to translate as no counterpart exists in other languages, and these must be addressed using context. In this case, the term is translated into *loser*, which is an acceptable translation considering the lack of better English terms.

The second row shows an example where multiple lexical variants in English are used. In this example, the Chinese translation is translated formally without the stylistic features present in the English sentence. However, in some parallel sentences, these properties are preserved during translation. For instance, the term *toooooo* in the third row corresponds to the word *too* with extra *o*'s to further emphasize the following adjective *thick*. Likewise, in the Chinese sentence, it is translated into 太*XX*, which is composed by the character 太, an equivalent for the word *too* in English, and the string *XX*, which adds a similar connotation to the extra *o*'s in the English sentence.

Finally, in the fourth and fifth rows, we observe some examples of "Chinglish," where new English words are constructed from Chinese terms. In the fourth row, the term 牛逼, which is an adjective to describe that something is great, is translated into *niubility*, which is constructed by taking the Pinyin representation for the term 牛逼(*niubi*) and adding the English suffix *lity*. More examples are given in the fifth row, where other similar terms (niubility, zhuangbility, shability, and erbility) are also

1	It's said all the **losers** around the country have been downloading a 982.77M video over the past two or three days. The latest greetings are: "have you done?" and the answer is: "Not yet! It got stuck at 83%!" Quit playing dumb. You know what I'm talking about	据说这两天全国的**屌丝**都在下载一个大小为982.77MB的文件；最新的见面问候语是："下完了没？"回答："还早呢，卡在83%不动了"别装、你们都懂的
2	To DanielVeuleman **yea iknw imma** work on that	对DanielVeuleman说，**是的**，我**知道**，**我正在**向那方面努力
3	People say real men can read women like a book. But don't you think this book is way **toooooo** thick?	他们说真男人懂得像读书一样去读懂女人，可是这书也**太XX**厚了吧
4	What's **Niubility**? **Niubility** is being able to get 1 million followers even if you post nothing and follow nobody on Weibo. We call it Han-style Niubility!	啥是**牛逼**？**牛逼**就是你在微博上一条消息未发，一个人不关注，却能得到百万粉丝。我们称之为韩式**牛逼**。
5	Many people think they are full of **niubility** and like to play **zhuangbility**, which only reflect their **shability** and **erbility**.	很多人觉得自己很**牛B**特喜欢**装B**，其实就是个**傻B**和**2B**！

Figure 6
Examples of parallel sentences extracted from Sina Weibo. These examples were hand-extracted because they contain elements that are characteristic of this domain.

translated. In this case, we can also observe examples of abbreviations in the Chinese translations, where the character 逼 is replaced by the letter *B*, which is the first letter in the Pinyin representation of the character.

It is also important to note that mainstream translation systems, such as Google Translate[11] and Bing Translator[12] fail to translate these examples. For instance, the term *diaosi* in the first row is translated character-by-character, which leads to a translation with a very different meaning. Both abbreviations in English (e.g., *imma*) and in Chinese (e.g., 逼*B*) also tend to be mistranslated. These are either incorrectly translated phonetically for English words, or character-by-character for Chinese terms. Finally, we also tried translating "Chinglish" terms using online systems and found that these are either not translated or translated incorrectly.

The existence of these terms in microblogs and other informal domains justifies the need for the methods to find parallel data where the translations for those terms can be learned. As such, we believe that the method we propose in this work has the potential to substantially improve the state-of-the-art MT systems in these domains.

8. Conclusions

This work presented a method for finding and extracting translated content present in microblog messages. More specifically, it finds posts that contain translated

11 https://translate.google.com.
12 http://www.bing.com/translator/.

segments within the post. We applied this to extract content from Twitter and Sina Weibo.

We present a method to efficiently identify multilingual tweets, which is used to filter the initial set of tweets. We then present an intra-document alignment model, which finds the most likely parallel segments, language pair, and alignments for each tweet. Finally, each set of parallel segments is classified as parallel or non-parallel by training a max-entropy classifier. To test and show the validity of the various steps of the extraction process, we use a crowdsourcing approach where non-experts can be used to generate annotated data.

Using this approach, we show that a considerable amount of parallel content can be obtained from Twitter and Sina Weibo. Applied to machine translation, it is demonstrated empirically that the extracted corpora have the potential to substantially increase the quality of the translations in informal genres.

The corpora that were collected and the annotated gold standards are available at `http://www.cs.cmu.edu/~lingwang/microtopia/`.

Acknowledgments

This work was done during the doctoral research of Wang Ling at Carnegie Mellon University and Instituto Superior Técnico. It was supported by national funds through FCT (Fundação para a Ciência e a Tecnologia) with reference UID/CEC/50021/2013, project MT4M (CMUP-EPB/TIC/0026/2013), and Ph.D. grant SFRH/BD/33769/2009. The authors wish to thank the anonymous reviewers for many helpful comments.

References

Ambati, Vamshi 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*, CSLDAMT '10, pages 62–65, Stroudsburg, PA.

Ambati, Vamshi, 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.

Bergsma, Shane, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language identification for creating language-specific Twitter collections. In *Proceedings of the Second Workshop on Language in Social Media*, LSM '12, pages 65–74, Stroudsburg, PA.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Medicine Learning Research*, 3:993–1022.

Braune, Fabienne and Alexander Fraser. 2010. Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 81–89, Stroudsburg, PA.

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

Dyer, Chris, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of NAACL-HLT*, pages 644–648. Atlanta, GA.

Eck, Matthias, Yury Zemlyanskiy, Joy Zhang, and Alex Waibel. 2014. Extracting translation pairs from social network content. In *IWSLT '14: International Workshop on Spoken Language Translation*, pages 200–205. Lake Tahoe, CA.

Fukushima, Ken'ichi, Kenjiro Taura, and Takashi Chikayama. 2006. A fast and accurate method for detecting English–Japanese parallel texts. In *Proceedings of the Workshop on Multilingual Language Resources and Interoperability*, pages 60–67, Sydney.

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

Gimpel, Kevin, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA.

Gottron, Thomas and Nedim Lipka. 2010. A comparison of language identification approaches on short, query-style texts. In C. Gurrin, Y. He, G. Kazai, et al., editors, *Advances in Information Retrieval*. Springer, pages 611–614.

Greenhill, Simon J. 2011. Levenshtein distances fail to identify language relationships accurately. *Computational Linguistics*, 37(4):689–698.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Exploration Newsletter*, 11(1):10–18.

Han, Bo and Timothy Baldwin. 2011. Lexical normalization of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 368–378, Stroudsburg, PA.

Jehl, Laura, Felix Hieber, and Stefan Riezler. 2012. Twitter translation using translation-based crosslingual retrieval. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 410–421, Stroudsburg, PA.

Johnson, J. Howard, Joel Martin, George Foster, and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proceedings of EMNLP-CoNLL 07*, pages 967–975. Prague

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. 2003. 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, Morristown, NJ.

Kong, Lingpeng, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. 2014. A dependency parser for Tweets. In *Proceedings of EMNLP*, pages 1001–1013. Doha.

Li, Bo and Juan Liu. 2008. Mining Chinese–English parallel corpora from the web. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP)*, pages 847–852. Hyderabad.

Lin, Dekang, Shaojun Zhao, Benjamin Van Durme, and Marius Paşca. 2008. Mining parenthetical translations from the Web by word alignment. In *Proceedings of ACL-08: HLT*, pages 994–1002, Columbus, OH.

Ling, Wang, Chris Dyer, Alan W. Black, and Isabel Trancoso. 2013. Paraphrasing 4 microblog normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 73–84, Seattle, WA.

Ling, Wang, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon.

Ling, Wang, Luis Marujo, Chris Dyer, Alan Black, and Isabel Trancoso. 2014. Crowdsourcing high-quality parallel data extraction from Twitter. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, WMT '14, pages 426–436, Stroudsburg, PA.

Ling, Wang, Guang Xiang, Chris Dyer, Alan Black, and Isabel Trancoso. 2013. Microblogs as parallel corpora. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 176–186, Sofia.

Lui, Marco, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40.

Marwick, Alice E. and Danah Boyd. 2010. I tweet honestly, I tweet passionately: Twitter users, context collapse, and the imagined audience. *New Media & Society*, 13(1):114–133.

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

Munteanu, Dragos Stefan, Alexander Fraser, and Daniel Marcu. 2004. Improved machine translation performance via parallel sentence extraction from

comparable corpora. In *Proceedings of HLT-NAACL*, pages 265–272, Boston, MA.

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

Owoputi, Olutobi, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL*, pages 380–389, Atlanta, GA.

Papineni, Kishore, 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 of the Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA.

Peng, Nanyun, Yiming Wang, and Mark Dredze. 2014. Learning polylingual topic models from code-switched social media documents. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, Volume 2: Short Papers*, pages 674–679. Baltimore, MD.

Post, Matt, 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.

Resnik, Philip and Noah A. Smith. 2003. The Web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Ritter, Alan, Mausam, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from Twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1104–1112, New York, NY.

Smith, Jason R., 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.

Ture, Ferhan and Jimmy Lin. 2012. Why not grab a free lunch? Mining large corpora for parallel sentences to improve translation modeling. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 626–630, Montréal.

Uszkoreit, Jakob, Jay M. Ponte, Ashok C. Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1101–1109. Beijing.

Wang, William Yang, Lingpeng Kong, Kathryn Mazaitis, and William W. Cohen. 2014. Dependency parsing for Weibo: An efficient probabilistic logic programming approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1152–1158, Doha.

Xu, Jia, Richard Zens, and Hermann Ney. 2005. Sentence segmentation using IBM word alignment model 1. In *Proceedings of EAMT 2005, 10th Annual Conference of the European Association for Machine Translation*, pages 280–287. Budapest.

Xu, Jinxi, Ralph Weischedel, and Chanh Nguyen. 2001. Evaluating a probabilistic model for cross-lingual information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 105–110, New York, NY.

Zaidan, Omar F. 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.

Zissman, Marc A. 1996. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on Speech and Audio Processing*, 4(13):31–44.