Are Two Heads Better than One? Crowdsourced Translation via a Two-Step Collaboration of Non-Professional Translators and Editors

Rui Yan, Mingkun Gao, Ellie Pavlick, and Chris Callison-Burch
Computer and Information Science Department,
University of Pennsylvania, Philadelphia, PA 19104, U.S.A.
{ruiyan,gmingkun,epavlick}@seas.upenn.edu, ccb@cis.upenn.edu

Abstract

Crowdsourcing is a viable mechanism for creating training data for machine translation. It provides a low cost, fast turnaround way of processing large volumes of data. However, when compared to professional translation, naive collection of translations from non-professionals yields low-quality results. Careful quality control is necessary for crowdsourcing to work well. In this paper, we examine the challenges of a two-step collaboration process with translation and post-editing by non-professionals. We develop graph-based ranking models that automatically select the best output from multiple redundant versions of translations and edits, and improves translation quality closer to professionals.

1 Introduction

Statistical machine translation (SMT) systems are trained using bilingual sentence-aligned parallel corpora. Theoretically, SMT can be applied to any language pair, but in practice it produces the state-of-art results only for language pairs with ample training data, like English-Arabic, English-Chinese, French-English, etc. SMT gets stuck in a severe bottleneck for many minority or 'low resource' languages with insufficient data. This drastically limits which languages SMT can be successfully applied to. Because of this, collecting parallel corpora for minor languages has become an interesting research challenge. There are various options for creating training data for new language pairs. Past approaches have examined harvesting translated documents from the web (Resnik and Smith, 2003; Uszkoreit et al., 2010; Smith et al., 2013), or discovering parallel fragments from comparable corpora (Munteanu and Marcu, 2005; Abdul-Rauf and Schwenk, 2009; Smith et al., 2010). Until relatively recently, little consideration has been given to creating parallel data from scratch. This is because the cost of hiring professional translators is prohibitively high. For instance, Germann (2001) hoped to hire professional translators to create a modest sized 100,000 word Tamil-English parallel corpus, but were stymied by the costs and the difficulty of finding good translators for a short-term commitment.

Recently, crowdsourcing has opened the possibility of translating large amounts of text at low cost using non-professional translators. Facebook localized its web site into different languages using volunteers (TechCrunch, 2008). DuoLingo turns translation into an educational game, and translates web content using its language learners (von Ahn, 2013).

Rather than relying on volunteers or gamification, NLP research into crowdsourcing translation has focused on hiring workers on the Amazon Mechanical Turk (MTurk) platform (Callison-Burch, 2009). This setup presents unique challenges, since it typically involves non-professional translators whose language skills are varied, and since it sometimes involves participants who try to cheat to get the small financial reward (Zaidan and Callison-Burch, 2011). A natural approach for trying to shore up the skills of weak bilinguals is to pair them with a native speaker of the target language to edit their translations. We review relevant research from NLP and human-computer interaction (HCI) on collaborative translation processes in Section 2. To sort good translations from bad, researchers often solicit multiple, redundant translations and then build models to try to predict which translations are the best, or which translators tend to produce the highest quality translations.

The contributions of this paper are:

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 1134–1144,
Baltimore, Maryland, USA, June 23-25 2014.

- An analysis of the difficulties posed by a two-step collaboration between editors and translators in Mechanical Turk-style crowdsourcing environments. Editors vary in quality, and poor editing can be difficult to detect.

- A new graph-based algorithm for selecting the best translation among multiple translations of the same input. This method takes into account the collaborative relationship between the translators and the editors.

2 Related work

In the HCI community, several researchers have proposed protocols for collaborative translation efforts (Morita and Ishida, 2009b; Morita and Ishida, 2009a; Hu, 2009; Hu et al., 2010). These have focused on an iterative collaboration between monolingual speakers of the two languages, facilitated with a machine translation system. These studies are similar to ours in that they rely on native speakers' understanding of the target language to correct the disfluencies in poor translations. In our setup the poor translations are produced by bilingual individuals who are weak in the target language, and in their experiments the translations are the output of a machine translation system.[1]

Another significant difference is that the HCI studies assume cooperative participants. For instance, Hu et al. (2010) recruited volunteers from the International Children's Digital Library (Hourcade et al., 2003) who were all well intentioned and participated out a sense of altruism and to build a good reputation among the other volunteer translators at `childrenslibrary.org`. Our setup uses anonymous crowd workers hired on Mechanical Turk, whose motivation to participate is financial. Bernstein et al. (2010) characterized the problems with hiring editors via MTurk for a word processing application. Workers were either lazy (meaning they made only minimal edits) or overly zealous (meaning they made many unnecessary edits). Bernstein et al. (2010) addressed this problem with a three step find-fix-verify process. In the first step, workers click on one word or phrase that needed to be corrected. In the next step, a separate group of workers proposed corrections to problematic regions that had been identified by multiple workers in the first pass. In the final step, other workers would validate whether the proposed corrections were good.

Most NLP research into crowdsourcing has focused on Mechanical Turk, following pioneering work by Snow et al. (2008) who showed that the platform was a viable way of collecting data for a wide variety of NLP tasks at low cost and in large volumes. They further showed that non-expert annotations are similar to expert annotations when many non-expert labelings for the same input are aggregated, through simple voting or through weighting votes based on how closely non-experts matched experts on a small amount of calibration data. MTurk has subsequently been widely adopted by the NLP community and used for an extensive range of speech and language applications (Callison-Burch and Dredze, 2010).

Although hiring professional translators to create bilingual training data for machine translation systems has been deemed infeasible, Mechanical Turk has provided a low cost way of creating large volumes of translations (Callison-Burch, 2009; Ambati and Vogel, 2010). For instance, Zbib et al. (2012; Zbib et al. (2013) translated 1.5 million words of Levine Arabic and Egyptian Arabic, and showed that a statistical translation system trained on the dialect data outperformed a system trained on 100 times more MSA data. Post et al. (2012) used MTurk to create parallel corpora for six Indian languages for less than $0.01 per word. MTurk workers translated more than half a million words worth of Malayalam in less than a week. Several researchers have examined the use of active learning to further reduce the cost of translation (Ambati et al., 2010; Ambati, 2012; Bloodgood and Callison-Burch, 2010). Crowdsourcing allowed real studies to be conducted whereas most past active learning were simulated. Pavlick et al. (2014) conducted a large-scale demographic study of the languages spoken by workers on MTurk by translating 10,000 words in each of 100 languages. Chen and Dolan (2012) examined the steps necessary to build a persistent multilingual workforce on MTurk.

This paper is most closely related to previous work by Zaidan and Callison-Burch (2011), who showed that non-professional translators could approach the level of professional translators. They solicited multiple redundant translations from dif-

[1] A variety of HCI and NLP studies have confirmed the efficacy of monolingual or bilingual individuals post-editing of machine translation output (Callison-Burch, 2005; Koehn, 2010; Green et al., 2013). Past NLP work has also examined automatic post-editing(Knight and Chander, 1994).

Urdu translator:
According to the territory's people the pamphlets from the Taaliban had been read in the announcements in all the mosques of the Northern Wazeerastan.
English post-editor:
According to locals, the pamphlet released by the Taliban was read out on the loudspeakers of all the mosques in North Waziristan.
LDC professional:
According to the local people, the Taliban's pamphlet was read over the loudspeakers of all mosques in North Waziristan.

Table 1: Different versions of translations.

ferent Turkers for a collection of Urdu sentences that had been previously professionally translated by the Linguistics Data Consortium. They built a model to try to predict on a sentence-by-sentence and Turker-by-Turker which was the best translation or translator. They also hired US-based Turkers to edit the translations, since the translators were largely based in Pakistan and exhibited errors that are characteristic of speakers of English as a language. Zaidan and Callison-Burch (2011) observed only modest improvements when incorporating these edited translation into their model. We attempt to analyze why this is, and we proposed a new model to try to better leverage their data.

3 Crowdsourcing Translation

Setup We conduct our experiments using the data collected by Zaidan and Callison-Burch (2011). This data set consists 1,792 Urdu sentences from a variety of news and online sources, each paired with English translations provided by non-professional translators on Mechanical Turk.

Each Urdu sentence was translated redundantly by 3 distinct translators, and each translation was edited by 3 separate (native English-speaking) editors to correct for grammatical and stylistic errors. In total, this gives us 12 non-professional English candidate sentences (3 unedited, 9 edited) per original Urdu sentence. 52 different Turkers took part in the translation task, each translating 138 sentences on average. In the editing task, 320 Turkers participated, averaging 56 sentences each. For comparison, the data also includes 4 different reference translations for each source sentence, produced by professional translators.

Table 1 gives an example of an unedited translation, an edited translation, and a professional translation for the same sentence. The translations provided by translators on MTurk are generally done conscientiously, preserving the meaning of the source sentence, but typically contain simple mistakes like misspellings, typos, and awkward word choice. English-speaking editors, despite having no knowledge of the source language, are able to fix these errors. In this work, we show that the collaboration design of two heads– non-professional Urdu translators and non-professional English editors– yields better translated output than would either one working in isolation, and can better approximate the quality of professional translators.

Analysis We know from inspection that translations seem to improve with editing (Table 1). Given the data from MTurk, we explore whether this is the case in general: Do all translations improve with editing? To what extent does the individual translator and the individual editor effect the quality of the final sentence?

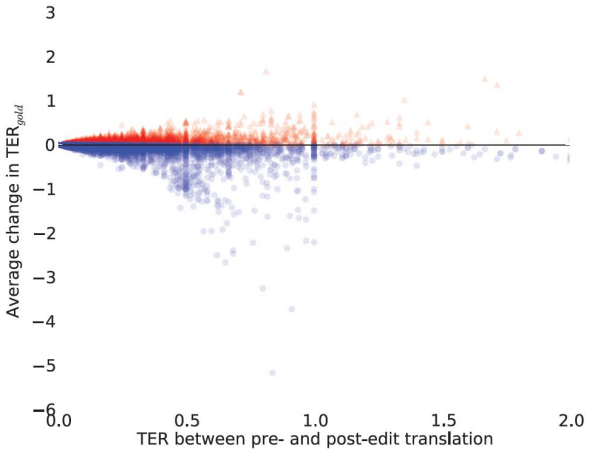

Figure 1: Relationship between editor aggressiveness and effectiveness. Each point represents an editor/translation pair. Aggressiveness (x-axis) is measured as the TER between the pre-edit and post-edit version of the translation, and effectiveness (y-axis) is measured as the average amount by which the editing reduces the translation's TER$_{gold}$. While many editors make only a few changes, those who make many changes can bring the translation substantially closer to professional quality.

We use *translation edit rate* (TER) as a measure of translation similarity. TER represents the amount of change necessary to transform one sentence into another, so a low TER means the two

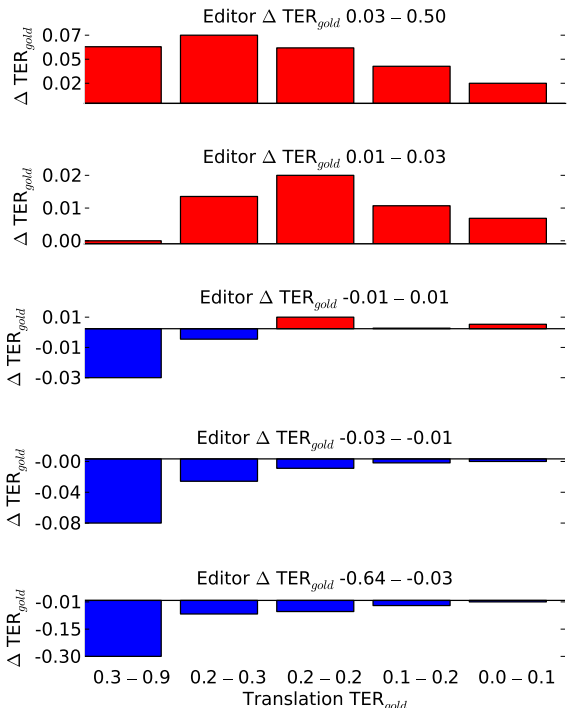

Figure 2: Effect of editing on translations of varying quality. Rows reflect bins of editors, with the worse editors (those whose changes result in increased TER$_{gold}$) on the top and the most effective editors (those whose changes result in the largest reduction in TER$_{gold}$) on the bottom. Bars reflect bins of translations, with the highest TER$_{gold}$ translations on the left, and the lowest on the right. We can see from the consistently negative Δ TER$_{gold}$ in the bottom row that good editors are able to improve both good and bad translations.

sentences are very similar. To capture the quality ("professionalness") of a translation, we take the average TER of the translation against each of our gold translations. That is, we define TER$_{gold}$ of translation t as

$$TER_{gold} = \frac{1}{4}\sum_{i=1}^{4} TER(gold_i, t) \qquad (1)$$

where a lower TER$_{gold}$ is indicative of a higher quality (more professional-sounding) translation.

We first look at editors along two dimensions: their aggressiveness and their effectiveness. Some editors may be very aggressive (they make many changes to the original translation) but still be ineffective (they fail to bring the quality of the translation closer to that of a professional). We measure aggressiveness by looking at the TER between the pre- and post-edited versions of each editor's translations; higher TER implies more aggressive editing. To measure effectiveness, we look at the change in TER$_{gold}$ that results from the editing; negative ΔTER$_{gold}$ means the editor effectively improved the quality of the translation, while positive ΔTER$_{gold}$ means the editing actually brought the translation further from our gold standard.

Figure 1 shows the relationship between these two qualities for individual editor/translation pairs. We see that while most translations require only a few edits, there are a large number of translations which improve substantially after heavy editing. This trend conforms to our intuition that editing is most useful when the translation has much room for improvement, and opens the question of whether good editors can offer improvements to translations of all qualities.

To address this question, we split our translations into 5 bins, based on their TER$_{gold}$. We also split our editors into 5 bins, based on their effectiveness (i.e. the average amount by which their editing reduces TER$_{gold}$). Figure 2 shows the degree to which editors at each level are able to improve the translations from each bin. We see that good editors are able to make improvements to translations of all qualities, but that good editing has the greatest impact on lower quality translations. This result suggests that finding good editor/translator pairs, rather than good editors and good translators in isolation, should produce the best translations overall. Figure 3 gives an example of how an initially medium-quality translation, when combined with good editing, produces a better result than the higher-quality translation paired with mediocre editing.

4 Problem Formulation

The problem definition of the crowdsourcing translation task is straightforward: given a set of candidate translations for a source sentence, we want to choose the best output translation.

This output translation is the result of the combined translation and editing stages. Therefore, our method operates over a heterogeneous network that includes translators and post-editors as well as the translated sentences that they produce. We frame the problem as follows. We form two graphs: the first graph (G_T) represents Turkers (translator/editor pairs) as nodes; the second graph (G_C) represents candidate translated and

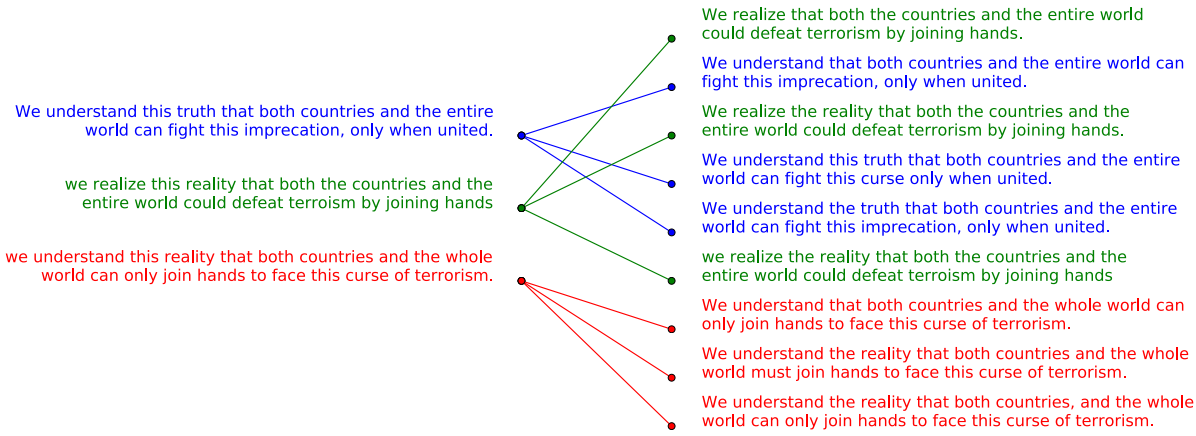

Figure 3: Three alternative translations (left) and the edited versions of each (right). Each edit on the right was produced by a different editor. Order reflects the TER_{gold} of each translation, with the lowest TER_{gold} on the top. Some translators receive low TER_{gold} scores due to superficial errors, which can be easily improved through editing. In the above example, the middle-ranked translation (green) becomes the best translation after being revised by a good editor.

post-edited sentences (henceforth "candidates") as nodes. These two graphs, G_T and G_C are combined as subgraphs of a third graph (G_{TC}). Edges in G_{TC} connect author pairs (nodes in G_T) to the candidate that they produced (nodes in G_C). Together, G_T, G_C, and G_{TC} define a co-ranking problem (Yan et al., 2012a; Yan et al., 2011b; Yan et al., 2012b) with linkage establishment (Yan et al., 2011a; Yan et al., 2012c), which we define formally as follows.

Let G denote the heterogeneous graph with nodes V and edges E. Let $G = (V,E) = (V_T, V_C, E_T, E_C, E_{TC})$. G is divided into three subgraphs, G_T, G_C, and G_{TC}. $G_C = (V_C, E_C)$ is a weighted undirected graph representing the candidates and their lexical relationships to one another. Let V_C denote a collection of translated and edited candidates, and E_C the lexical similarity between the candidates (see Section 4.3 for details). $G_T = (V_T, E_T)$ is a weighted undirected graph representing collaborations between Turkers. V_T is the set of translator/editor pairs. Edges E_T connect translator/editor pairs in V_T which share a translator and/or editor. Each collaboration (i.e. each node in V_T) produces a candidate (i.e. a node in V_C). $G_{TC} = (V_{TC}, E_{TC})$ is an unweighted bipartite graph that ties G_T and G_C together and represents "authorship". The graph G consists of nodes $V_{TC} = V_T \cup V_C$ and edges E_{TC} connecting each candidate with its authoring translator/post-editor pair. The three sub-networks (G_T, G_C, and G_{TC}) are illustrated in Figure 4.

4.1 Inter-Graph Ranking

The framework includes three random walks, one on G_T, one on G_C and one on G_{TC}. A random walk on a graph is a Markov chain, its states being the vertices of the graph. It can be described by a stochastic square matrix, where the dimension is the number of vertices in the graph, and the entries describe the transition probabilities from one vertex to the next. The mutual reinforcement framework couples the two random walks on G_T and G_C that rank candidates and Turkers in isolation. The ranking method allows us to obtain a global ranking by taking into account the intra-/inter-component dependencies. In the following sections, we describe how we obtain the rankings on G_T and G_C, and then move on to discuss how the two are coupled.

Our algorithm aims to capture the following intuitions. A candidate is important if 1) it is similar to many of the other proposed candidates and 2) it is authored by better qualified translators and/or post-editors. Analogously, a translator/editor pair is believed to be better qualified if 1) the editor is collaborating with a good translator and vice versa and 2) the pair has authored important candidates. This ranking schema is actually a reinforced process across the heterogeneous graphs. We use two vectors $\mathbf{c} = [\pi(c)]_{|c|\times 1}$ and $\mathbf{t} = [\pi(t)]_{|t|\times 1}$ to denote the saliency scores $\pi(.)$ of candidates and Turker pairs. The above-mentioned intuitions can be formulated as follows:

- **Homogeneity.** We use adjacency matrix

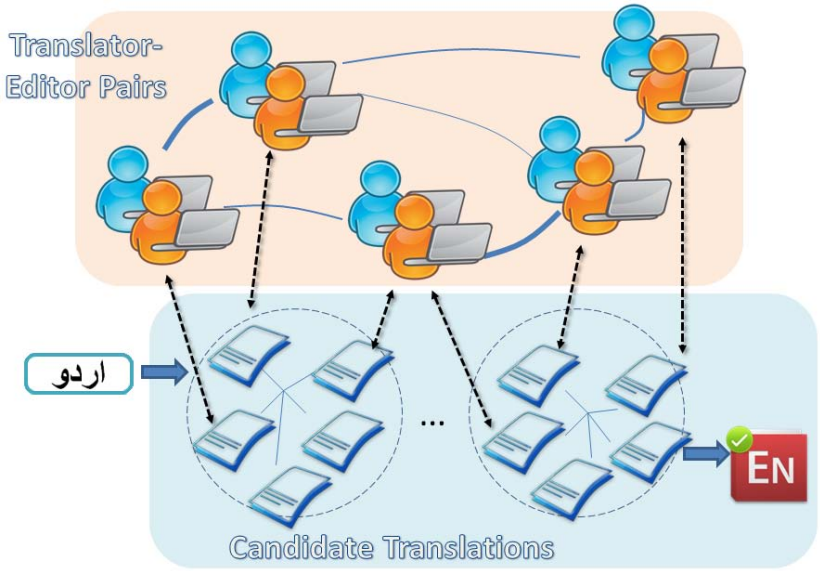

Figure 4: 2-step collaborative crowdsourcing translation model based on graph ranking framework including three sub-networks. The undirected links between users denotes *translation-editing* collaboration. The undirected links between candidate translations indicate lexical similarity between candidates. A bipartite graph ties candidate and Turker networks together by authorship (to make the figure clearer, some linkage is omitted). A dashed circle indicates the group of candidate translations for a single source sentence to translate.

$[M]_{|\mathbf{c}|\times|\mathbf{c}|}$ to describe the homogeneous affinity between candidates and $[N]_{|\mathbf{t}|\times|\mathbf{t}|}$ to describe the affinity between Turkers.

$$\mathbf{c} \propto M^T\mathbf{c}, \qquad \mathbf{t} \propto N^T\mathbf{t} \tag{2}$$

where $c = |V_C|$ is the number of vertices in the candidate graph and $t = |V_T|$ is the number of vertices in the Turker graph. The adjacency matrix $[M]$ denotes the transition probabilities between candidates, and analogously matrix $[N]$ denotes the affinity between Turker collaboration pairs.

• **Heterogeneity.** We use an adjacency matrix $[\hat{W}]_{|\mathbf{c}|\times|\mathbf{t}|}$ and $[\bar{W}]_{|\mathbf{t}|\times|\mathbf{c}|}$ to describe the authorship between the output candidate and the producer Turker pair from both of the candidate-to-Turker and Turker-to-candidate perspectives.

$$\mathbf{c} \propto \hat{W}^T\mathbf{t}, \qquad \mathbf{t} \propto \bar{W}^T\mathbf{c} \tag{3}$$

All affinity matrices will be defined in the next section. By fusing the above equations, we can have the following iterative calculation in matrix forms. For numerical computation of the saliency scores, the initial scores of all sentences and Turkers are set to 1 and the following two steps are alternated until convergence to select the best candidate.

Step 1: compute the saliency scores of candidates, and then normalize using ℓ-1 norm.

$$\begin{aligned}\mathbf{c}^{(n)} &= (1-\lambda)M^T\mathbf{c}^{(n-1)} + \lambda\hat{W}\mathbf{t}^{(n-1)} \\ \mathbf{c}^{(n)} &= \mathbf{c}^{(n)}/||\mathbf{c}^{(n)}||_1\end{aligned} \tag{4}$$

Step 2: compute the saliency scores of Turker pairs, and then normalize using ℓ-1 norm.

$$\begin{aligned}\mathbf{t}^{(n)} &= (1-\lambda)N^T\mathbf{t}^{(n-1)} + \lambda\bar{W}\mathbf{c}^{(n-1)} \\ \mathbf{t}^{(n)} &= \mathbf{t}^{(n)}/||\mathbf{t}^{(n)}||_1\end{aligned} \tag{5}$$

where λ specifies the relative contributions to the saliency score trade-off between the homogeneous affinity and the heterogeneous affinity. In order to guarantee the convergence of the iterative form, we must force the transition matrix to be stochastic and irreducible. To this end, we must make the $\mathbf{c}$ and $\mathbf{t}$ *column stochastic* (Langville and Meyer, 2004). $\mathbf{c}$ and $\mathbf{t}$ are therefore normalized after each iteration of Equation (4) and (5).

4.2 Intra-Graph Ranking

The standard PageRank algorithm starts from an arbitrary node and randomly selects to either follow a random out-going edge (considering the weighted transition matrix) or to jump to a random node (treating all nodes with equal probability).

In a simple random walk, it is assumed that all nodes in the transitional matrix are equi-probable before the walk starts. Then **c** and **t** are calculated as:

$$\mathbf{c} = \mu M^T \mathbf{c} + (1-\mu)\frac{\mathbf{1}}{|V_C|} \quad (6)$$

and

$$\mathbf{t} = \mu N^T \mathbf{t} + (1-\mu)\frac{\mathbf{1}}{|V_T|} \quad (7)$$

where **1** is a vector with all elements equaling to 1 and the size is correspondent to the size of V_C or V_T. μ is the damping factor usually set to 0.85, as in the PageRank algorithm.

4.3 Affinity Matrix Establishment

We introduce the affinity matrix calculation, including homogeneous affinity (i.e., M, N) and heterogeneous affinity (i.e., $\hat{W}, \bar{W}$).

As discussed, we model the collection of candidates as a weighted undirected graph, G_C, in which nodes in the graph represent candidate sentences and edges represent lexical relatedness. We define an edge's weight to be the cosine similarity between the candidates represented by the nodes that it connects. The adjacency matrix M describes such a graph, with each entry corresponding to the weight of an edge.

$$\begin{aligned} \mathcal{F}(c_i, c_j) &= \frac{c_i \cdot c_j}{||c_i||||c_j||} \\ M_{ij} &= \frac{\mathcal{F}(c_i, c_j)}{\sum_k \mathcal{F}(c_i, c_k)} \end{aligned} \quad (8)$$

where $\mathcal{F}(.)$ is the cosine similarity and c is a term vector corresponding to a candidate. We treat a candidate as a short document and weight each term with *tf.idf* (Manning et al., 2008), where *tf* is the term frequency and *idf* is the inverse document frequency.

The Turker graph, G_T, is an undirected graph whose edges represent "collaboration." Formally, let t_i and t_j be two translator/editor pairs; we say that pair t_i "collaborates with" pair t_j (and therefore, there is an edge between t_i and t_j) if t_i and t_j share either a translator or an editor (or share both a translator and an editor). Let the function $\mathcal{I}(t_i, t_j)$ denote the number of "collaborations" ($\#col$) between t_i and t_j.

$$\mathcal{I}(t_i, t_j) = \begin{cases} \#col & (e_{ij} \in E_T) \\ 0 & \text{otherwise} \end{cases}, \quad (9)$$

Then the adjacency matrix N is then defined as

$$N_{ij} = \frac{\mathcal{I}(t_i, t_j)}{\sum_k \mathcal{I}(t_i, t_k)} \quad (10)$$

In the bipartite candidate-Turker graph G_{TC}, the entry $E_{TC}(i, j)$ is an indicator function denoting whether the candidate c_i is generated by t_j:

$$\mathcal{A}(c_i, t_j) = \begin{cases} 1 & (e_{ij} \in E_{TC}) \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

Through E_{TC} we define the weight matrices $\bar{W}_{ij}$ and $\hat{W}_{ij}$, containing the conditional probabilities of transitions from c_i to t_j and vice versa:

$$\begin{aligned} \bar{W}_{ij} &= \frac{\mathcal{A}(c_i, t_j)}{\sum_k \mathcal{A}(c_i, t_k)}, \\ \hat{W}_{ij} &= \frac{\mathcal{A}(c_i, t_j)}{\sum_k \mathcal{A}(c_k, t_j)} \end{aligned} \quad (12)$$

5 Evaluation

We are interested in testing our random walk method, which incorporates information from both the candidate translations and from the Turkers. We want to test two versions of our proposed collaborative co-ranking method: 1) based on the unedited translations only and 2) based on the edited sentences after translator/editor collaborations.

Metric Since we have four professional translation sets, we can calculate the Bilingual Evaluation Understudy (BLEU) score (Papineni et al., 2002) for one professional translator (P1) using the other three (P2,3,4) as a reference set. We repeat the process four times, scoring each professional translator against the others, to calculate the expected range of professional quality translation. In the following sections, we evaluate each of our methods by calculating BLEU scores against the same four sets of three reference translations. Therefore, each number reported in our experimental results is an average of four numbers, corresponding to the four possible ways of choosing 3 of the 4 reference sets. This allows us to compare the BLEU score achieved by our methods against the BLEU scores achievable by professional translators.

Baselines As a naive baseline, we choose one candidate translation at random for each input Urdu sentence. To establish an upper bound for our methods, and to determine if there exist high-quality Turker translations at all, we compute four

Reference (Avg.)	42.51
Oracle (Seg-Trans)	44.93
Oracle (Seg-Trans+Edit)	48.44
Oracle (Turker-Trans)	38.66
Oracle (Turker-Trans+Edit)	39.16
Random	30.52
Lowest TER	35.78
Graph Ranking (Trans)	38.88
Graph Ranking (Trans+Edit)	**41.43**

Table 2: Overall BLEU performance for all methods (with and without post-editing). The highlighted result indicates the best performance, which is based on both candidate sentences and Turker information.

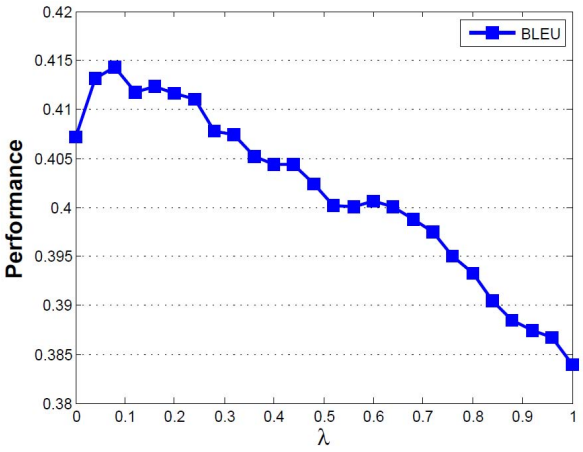

Figure 5: Effect of candidate-Turker coupling (λ) on BLEU score.

oracle scores. The first oracle operates at the segment level on the sentences produced by translators only: for each source segment, we choose from the translations the one that scores highest (in terms of BLEU) against the reference sentences. The second oracle is applied similarly, but chooses from the candidates produced by the collaboration of translator/post-editor pairs. The third oracle operates at the worker level: for each source segment, we choose from the translations the one provided by the worker whose translations (over all sentences) score the highest on average. The fourth oracle also operates at the worker level, but selects from sentences produced by translator/post-editor collaborations. These oracle methods represent ideal solutions under our scenario. We also examine two voting-inspired methods. The first method selects the translation with the minimum average TER (Snover et al., 2006) against the other translations; intuitively, this would represent the "consensus" translation. The second method selects the translation generated by the Turker who, on average, provides translations with the minimum average TER.

Results A summary of our results in given in Table 2. As expected, random selection yields bad performance, with a BLEU score of 30.52. The oracles indicate that there is usually an acceptable translation from the Turkers for any given sentence. Since the oracles select from a small group of only 4 translations per source segment, they are not overly optimistic, and rather reflect the true potential of the collected translations. On average, the reference translations give a score of 42.38. To put this in perspective, the output of a state-of-the-art machine translation system (the syntax-based variant of Joshua) achieves a score of 26.91, which is reported in (Zaidan and Callison-Burch, 2011). The approach which selects the translations with the minimum average TER (Snover et al., 2006) against the other three translations (the "consensus" translation) achieves BLEU scores of 35.78.

Using the raw translations without post-editing, our graph-based ranking method achieves a BLEU score of 38.89, compared to Zaidan and Callison-Burch (2011)' s reported score of 28.13, which they achieved using a linear feature-based classification. Their linear classifier achieved a reported score of 39.06[2] when combining information from both translators and editors. In contrast, our proposed graph-based ranking framework achieves a score of 41.43 when using the same information. This boost in BLEU score confirms our intuition that the hidden collaboration networks between candidate translations and transltor/editor pairs are indeed useful.

Parameter Tuning There are two parameters in our experimental setups: μ controls the probability of starting a new random walk and λ controls the coupling between the candidate and Turker subgraphs. We set the damping factor μ to 0.85, following the standard PageRank paradigm. In order to determine a value for λ, we used the average BLEU, computed against the professional refer-

[2]Note that the data we used in our experiments are slightly different, by discarding nearly 100 NULL sentences in the raw data. We do not re-implement this baseline but report the results from the paper directly. According to our experiments, most of the results generated by baselines and oracles are very close to the previously reported values.

Plain ranking	38.89
w/o collaboration	38.88
Shared translator	41.38
Shared post-editor	41.29
Shared Turker	**41.43**

Table 3: Variations of all component settings.

ence translations, as a tuning metric. We experimented with values of λ ranging from 0 to 1, with a step size of 0.05 (Figure 5). Small λ values place little emphasis on the candidate/Turker coupling, whereas larger values rely more heavily on the co-ranking. Overall, we observed better performance with values within the range of 0.05-0.15. This suggests that both sources of information– the candidate itself and its authors– are important for the crowdsourcing translation task. In all of our reported results, we used the $\lambda = 0.1$.

Analysis We examine the relative contribution of each component of our approach on the overall performance. We first examine the centroid-based ranking on the candidate sub-graph (G_C) alone to see the effect of voting among translated sentences; we denote this strategy as *plain ranking*. Then we incorporate the standard random walk on the Turker graph (G_T) to include the structural information but without yet including any collaboration information; that is, we incorporate information from G_T and G_C without including edges linking the two together. The co-ranking paradigm is exactly the same as the framework described in Section 3.2, but with simplified structures.

Finally, we examine the two-step collaboration based candidate-Turker graph using several variations on edge establishment. As before, the nodes are the translator/post-editor working pairs. We investigate three settings in which 1) edges connect two nodes when they share only a translator, 2) edges connect two nodes when they share only a post-editor, and 3) edges connect two nodes when they share either a translator or a post-editor. These results are summarized in Table 3.

Interestingly, we observe that when modeling the linkage between the collaboration pairs, connecting Turker pairs which share either a translator or the post-editor achieves better performance than connecting pairs that share only translators or connecting pairs which share only editors. This result supports the intuition that a denser collaboration matrix will help propagate saliency to good translators/post-editors and hence provides better predictions for candidate quality.

6 Conclusion

We have proposed an algorithm for using a two-step collaboration between non-professional translators and post-editors to obtain professional-quality translations. Our method, based on a co-ranking model, selects the best crowdsourced translation from a set of candidates, and is capable of selecting translations which near professional quality.

Crowdsourcing can play a pivotal role in future efforts to create parallel translation datasets. In addition to its benefits of cost and scalability, crowdsourcing provides access to languages that currently fall outside the scope of statistical machine translation research. In future work on crowdsourced translation, further benefits in quality improvement and cost reduction could stem from 1) building ground truth data sets based on high-quality Turkers' translations and 2) identifying when sufficient data has been collected for a given input, to avoid soliciting unnecessary redundant translations.

Acknowledgements

This material is based on research sponsored by a DARPA Computer Science Study Panel phase 3 award entitled "Crowdsourcing Translation" (contract D12PC00368). The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements by DARPA or the U.S. Government. This research was supported by the Johns Hopkins University Human Language Technology Center of Excellence and through gifts from Microsoft, Google and Facebook.

References

Sadaf Abdul-Rauf and Holger Schwenk. 2009. On the use of comparable corpora to improve SMT performance. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 16–23, March.

Vamshi Ambati and Stephan Vogel. 2010. Can crowds build parallel corpora for machine translation systems? In *Workshop on Creating Speech and Language Data with MTurk*.

Vamshi Ambati, Stephan Vogel, and Jaime G Carbonell. 2010. Active learning and crowd-sourcing for machine translation. In *LREC*, volume 11, pages 2169–2174. Citeseer.

Vamshi Ambati. 2012. *Active Learning and Crowdsourcing for Machine Translation in Low Resource Scenarios*. Ph.D. thesis, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Michael S. Bernstein, Greg Little, Robert C. Miller, Bjrn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. 2010. Soylent: a word processor with a crowd inside. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST)*.

Michael Bloodgood and Chris Callison-Burch. 2010. Large-scale cost-focused active learning for statistical machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12, June.

Chris Callison-Burch. 2005. Linear B system description for the 2005 NIST MT evaluation exercise. In *Proceedings of Machine Translation Evaluation Workshop*.

Chris Callison-Burch. 2009. Fast, cheap, and creative: Evaluating translation quality using amazon's mechanical turk. In *Proceedings of EMNLP*.

David L. Chen and William B. Dolan. 2012. Building a persistent workforce on mechanical turk for multilingual data collection. In *Proceedings of the Human Computer Interaction International Conference*.

Ulrich Germann. 2001. Building a statistical machine translation system from scratch: How much bang for the buck can we expect? In *Proceedings of the Workshop on Data-driven Methods in Machine Translation - Volume 14*, DMMT '01, pages 1–8.

Spence Green, Jeffrey Heer, and Christopher D. Manning. 2013. The efficacy of human post-editing for language translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 439–448.

Juan Pablo Hourcade, Benjamin B Bederson, Allison Druin, Anne Rose, Allison Farber, and Yoshifumi Takayama. 2003. The international children's digital library: viewing digital books online. *Interacting with Computers*, 15(2):151–167.

Chang Hu, Benjamin B. Bederson, and Philip Resnik. 2010. Translation by iterative collaboration between monolingual users. In *Proceedings of ACM SIGKDD Workshop on Human Computation (HCOMP)*.

Chang Hu, Benjamin B. Bederson, Philip Resnik, and Yakov Kronrod. 2011. Monotrans2: A new human computation system to support monolingual translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1133–1136.

Chang Hu. 2009. Collaborative translation by monolingual users. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, pages 3105–3108.

Martin Kay. 1998. The proper place of men and machines in language translation. *Machine Translation*, 12(1/2):3–23, January.

Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *In Proceedings of AAAI*.

Philipp Koehn. 2010. Enabling monolingual translators: Post-editing vs. options. In *HLT-NAACL'10*, pages 537–545, June.

Amy N Langville and Carl D Meyer. 2004. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Ann Irvine, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Ziyuan Wang, Jonathan Weese, and Omar Zaidan. 2010. Joshua 2.0: A toolkit for parsing-based machine translation with syntax, semirings, discriminative training and other goodies. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 133–137, July.

Annie Louis and Ani Nenkova. 2013. What makes writing great? first experiments on article quality prediction in the science journalism domain. *Transactions of Association for Computational Linguistics*.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

Daisuke Morita and Toru Ishida. 2009a. Collaborative translation by monolinguals with machine translators. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 361–366.

Daisuke Morita and Toru Ishida. 2009b. Designing protocols for collaborative translation. In *International Conference on Principles of Practice in Multi-Agent Systems (PRIMA-09)*, pages 17–32. Springer.

Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Comput. Linguist.*, 31(4):477–504, December.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318.

Ellie Pavlick, Matt Post, Ann Irvine, Dmitry Kachaev, and Chris Callison-Burch. 2014. The language demographics of Amazon Mechanical Turk. *Transactions of the Association for Computational Linguistics (TACL)*, 2(Feb):79–92.

Matt Post, Chris Callison-Burch, and Miles Osborne. 2012. Constructing parallel corpora for six Indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*.

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

Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *HLT-NAACL'10*, pages 403–411.

Jason Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. 2013. Dirt cheap web-scale parallel text from the Common Crawl. In *Proceedings of the 2013 Conference of the Association for Computational Linguistics (ACL 2013)*, July.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263.

TechCrunch. 2008. Facebook taps users to create translated versions of site, January.

Jakob Uszkoreit, 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*, COLING '10, pages 1101–1109.

Luis von Ahn. 2013. Duolingo: Learn a language for free while helping to translate the web. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces*, IUI '13, pages 1–2.

Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. 2011a. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 433–443.

Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. 2011b. Evolutionary timeline summarization: A balanced optimization framework via iterative substitution. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 745–754.

Rui Yan, Mirella Lapata, and Xiaoming Li. 2012a. Tweet recommendation with graph co-ranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 516–525.

Rui Yan, Xiaojun Wan, Mirella Lapata, Wayne Xin Zhao, Pu-Jen Cheng, and Xiaoming Li. 2012b. Visualizing timelines: Evolutionary summarization via iterative reinforcement between text and image streams. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 275–284.

Rui Yan, Zi Yuan, Xiaojun Wan, Yan Zhang, and Xiaoming Li. 2012c. Hierarchical graph summarization: leveraging hybrid information through visible and invisible linkage. In *PAKDD'12*, pages 97–108. Springer.

Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1220–1229.

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012. Machine translation of Arabic dialects. In *The 2012 Conference of the North American Chapter of the Association for Computational Linguistics*.

Rabih Zbib, Gretchen Markiewicz, Spyros Matsoukas, Richard Schwartz, and John Makhoul. 2013. Systematic comparison of professional and crowdsourced reference translations for machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia.

Xin Wayne Zhao, Yanwei Guo, Rui Yan, Yulan He, and Xiaoming Li. 2013. Timeline generation with social attention. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 1061–1064.