Instructions for ACL2012 Proceedings

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Abstract

This document contains the instructions for preparing a camera-ready manuscript for the proceedings of ACL2012. The document itself conforms to its own specifications, and is therefore an example of what your manuscript should look like. These instructions should be used for both papers submitted for review and for final versions of accepted papers. Authors are asked to conform to all the directions reported in this document.

1 Introduction

Crowdsourcing is a promising new mechanism for collecting large volumes of annotated data at low cost. Platforms like Amazon Mechanical Turk (MTurk) provide researchers with access to large groups of people, who can complete 'human intelligence tasks' that are beyond the scope of current artificial intelligence. Since statistical natural language processing benefits from increased amount of labeled training data, many NLP researchers have focused on creating speech and language data through crowdsourcing (for example, **?**; **?**) and others). One NLP application that has been the focus of crowdsourced data collection is statistical machine translation (SMT) which requires large bilingual sentence-aligned parallel corpora to train translation models. Crowdsourcing's low costs has made i possible to hire people to create sufficient volumes of translation in order to train SMT systems.

However, crowdsourcing is not perfect, and one of its most pressing challenges is how to ensure the quality of the data that is created by it. Unlike more traditional employment mechanism, where our annotator are pre-vetted and their skills are attested for, in crowdsourcing very little is known about the annotators. They are not professional translators, and their are no built-in mechanisms for testing their language skills. They complete tasks without any oversight. Thus, translations produced via crowdousrcing may be low quality . Previous work has addressed this problem, showing that non-professional translators hired on Amazon Mechanical Turk (MTurk) can achieve professional-level quality, by soliciting multiple translations of each source sentence and then choosing the best translation (Zaidan and Callison-Burch, 2011).

In this paper we focus on a different aspect of crowdsourcing than Zaidan and Callison-Burch (2011). We attempt to achieve the same high quality while **minimizing the associated costs**. We reduce costs using two complementary methods: (1) We quickly identify and filter out workers who produce low quality translations. The goal is to reduce the number of worker we hire, and retain only high quality translators. (2) Instead of soliciting a fixed number of translations for each foreign sentence, we stop soliciting translations after we get an acceptable one. We do so by building models to distinguish between acceptable translations and unacceptable ones. The goal is to reduces the number of independent translations that we solicit for each source sentence. Our work stands in contrast with Zaidan and Callison-Burch (2011) who had no model of annotator quality, and who always solicited and paid for a fixed number of translations of each source segment.

In this paper we demonstrate that

- Workers can be ranked by quality with high correlation against a gold standard ranking (ρ of 0.XXX), using logistic regression and a variety of features, or initially testing them using a small amount of calibration data with known professional translations.
- This ranking can be established after observing very small amounts of data (reaching ρ of 0.XXX after seeing only 10 translations from each worker), so bad workers can be filtered out quickly.
- Our models can predict whether a given translation is acceptable with high accuracy, substantially reducing the number of redundant translations needed for every source segment.
- We can achieve a similar BLEU score as Zaidan and Callison-Burch (2011) at $\frac{1}{X}$ of the cost.

2 Cost Optimization

There are two approaches that we use to reduce costs. In the first approach, we attempt to quickly rank workers and discard low ranking workers. In the second approach, we reduce the number of translations that we buy for each foreign sentence. After receiving a translation, we decide whether its quality is sufficient or whether we ought to pay for another translation (with the hope that the subsequent translation will be better). In both cases we aim to reduce costs, while keeping our overall translation quality high.

For the first approach, as mentioned in section 3.3, we show we are able to rank workers accurately using only the translations of their first 4 HITs. The ranked list of workers using 4 HITs is almost identical to the gold standard ranking that uses all HITs. In other words, we can predict workers' performance reasonably well, as long as we have obtained a small number of HITs from them. (Assuming that we have professional translations of their initial HITs, which we can use to calculate their translation quality). Consequently, we can decide whether to continue to hire a worker in a very short time after analyzing the first HIT(s) provided by each worker.

For the second approach, we train a model to decide whether a translation is 'good enough,' in which case we don't need to pay for another redundant translation of the source sentence. To perform this experiment, we divide the data into 3 parts: 10% of the data as a training set, 10% of the data as a validation set and the remaining 80% of the data as a test set. We train the model to score each translation we've got already, to use this score to evaluate whether to get another translation. The challenge is how to set the threshold to separate acceptable translations and unacceptable ones.

In our design, we set the threshold empirically using the validation set after we have trained the model on the disjoint training set. More specifically, during the training process, we get the lower bound and upper bound of scores for translations in the training set. Then we search for the threshold through traversing from the lower bound to the upper bound by a small step size. We use each value in the process as the potential threshold. We score translations of the foreign sentences in the validation set. Since this approach assumes a temporal ordering of the translations, we compute the scores for each translation of a source sentence using the time-ordering of when Turkers submitted them. There are 2 conditions on the halt of this process for each foreign sentence: 1) the predicted BLEU score of some translation (submitted earlier than the last translation) is higher than the threshold or 2) we have scored all 4 translations.

To evaluate the performance of the model running with different thresholds, we first compute an upper bound by selecting the best translation among all 4 candidates for each foreign sentence of the validation set according to our model. We call this set S_{upper}. S_{upper} is the highest BLEU score we can get by choosing translation using the model, since it has access to all of the available translations.

After we have used the validation set to sweep various threshold values, we can pick a suitable value for the threshold by picking the lowest value that is within some delta of S_{upper}, say 90%.

Finally, we retrain our model using the union set of the training set and validation set, use the resulting model on the test set. We evaluate the model's performance by counting the average number of candidate translations that it solicits per source sentence, and by computing the loss in overall BLEU score compared to when it had access to all 4 translations. This evaluation shows how much money our

model would save by eliminating unnecessary redundancy in the translation process, and how close it is to the upper bound on translation quality when using all of the translations from the original set.

3 Data

3.1 Data Collection

We use the data collected by Zaidan and Callison-Burch (2011) through Amazon's Mechanical Turk(MTurk). MTurk is an online platform provided to people for completing Human Intelligence Tasks(HIT) with a relatively low cost. We use their Urdu-to-English 2009 NIST Evaluation Set as our corpus. Zaidan and Callison-Burch (2011) translated the Urdu side to English through MTurk. They collected the translations in the unit of Human Intelligence Tasks(or HITs). In every HIT, they posted 10 Urdu sentences to be translated. Every sentence is translated by 4 workers, and subsequently post-edited by 10 additional workers.[1] This data set also has four corresponding professional translations for each of the Urdu sentences, collected by LDC. This makes it possible to compare the Turkers' translation quality to professionals.

3.2 Feature Extraction

Following Zaidan and Callison-Burch (2011), we extract a number of features from the translations and workers' self-reported language skills. We use these to build feature vectors used in tuning model and choosing the best translations from the candidates. POTENTIALLY: We extend Zaidan and Callison-Burch (2011)'s feature set to include additional bilingual features, which were not part of that original work.

Sentence-Level Features (9 Features)

This feature set contains language based features to solely implicate the quality of an English sentence without any suggestion on the bond of the meaning between the source sentence and the translation . This set of features tells good English sentences apart bad ones. The reason we use this set of features is that a good English sentence is the prerequisite of being a good English translation.

- Language model features: we assign a log probability and a per-word perplexity score for each sentence. We use SRILM toolkit to calculate perplexity score for each sentence based on 5-gram language model trained on English Gigaword corpus.

- Sentence length features: we use the ratio of the length of the Urdu source sentence to the length of the translation sentence as feature since a good translation is expected to be comparable in length with source sentence. We add two such ratio features(one is designed for unreasonably short translation and the other is for unreasonably long translation).

- Web *n*-gram log probability feature: we add the Web *n*-gram log probability feature to reflect the probability of the *n*-grams(up to length 5) exist in the Microsoft Web N-Gram Corpus. For short sentences whose length is less than 5, we use the sentence length as the order of the *n*-gram in calculation.

- Web *n*-gram geometric average features: we calculate the geometric average *n*-gram to evaluate the average matching over different *n*-grams. We add 3 features correspondent to max *n*-gram length of 3,4 and 5. Specifically, P_i denotes the log probability of *i*-gram and these 3 features are represented in $\sqrt[3]{P_1P_2P_3}$,$\sqrt[4]{P_1P_2P_3P_4}$ and $\sqrt[5]{P_1P_2P_3P_4P_5}$.

- Edit rate to other translations: In posterior methods, to minimize Bayes risk, we choose the translation that is most similar to other translations. Taking this into consideration, we add the edit rate feature to implement the similarity among all candidates translations.

Worker-Level Features (15 Features)

We take the quality of workers into consideration and add worker level features since the intuition that good workers can always high quality translations.

- Aggregate features: for each sentence level feature, we use the average values over all trans-

[1]Zaidan and Callison-Burch (2011) collected their translations in two batches. The first batch contained 1 translation, each with 1 post-edited version. The second contained an additional 3 translations, each of which was post-edited by 3 workers.

lations provided by the same worker as that worker's aggregate feature values.

- Language abilities: we collect worker's language ability information about whether the worker is a native Urdu speaker or native English speaker and how long they have spoken English or Urdu and add four features correspondent to the four aspects above.
- Worker Location: we add two features to indicate whether a worker is located in Pakistan or India.

Ranking Features (3 Features)

Zaidan and Callison-Burch (2011) collected 5 ranking labels for each translation and refine 3 features from these labels.

- Average Ranking: the average of the 5 ranking labels for this translation.
- Is-Best percentage: this feature shows how often a translation is ranked as the best translation among all candidates translation.
- Is-Better percentage: how often a translation is ranked as a better translation based on the pairwise comparisons.

Calibration Feature (1 Feature)

We use the average BLEU score of translations provided by the same worker as the calibration feature. The BLEU score is computed against references.

Word Alignment Feature (1 Feature)

We

3.3 Data Analysis

We analyze the data considering the timing information that accompanies the translations that we collected. In this analysis, we'd like to evaluate the performance of each worker as time goes on. Each worker translates one or more HITs. Since the translations were collected in two batches, which started at different times, we assign a relative time to each assignment to simulate what would have happened if both batches were run at the same time. For each HIT, we assign it a relative time by calculating difference between the HIT's submission time and the HIT's creation time. This gives the HIT's relative submission time. For each worker, we calculated the BLEU score for each of their HITs and analyze the translation quality of the worker at different time points. We are interested in seeing whether workers produce consistently good translations over time or if their quality drops off over time, and if workers who produce bad translations submit them more quickly than workers who submit good translations. Figure 1 illustrates workers' translation quality at across time. In this graph, each tick represent a single translation HIT, and depicts the HIT's BLEU score (color) and its size/number of sentences (thickness). The worker who provided the translation is graphed on the y axis, and the submission time is plotted on the x axis. We set the median of all HITs' BLEU scores as the threshold. If the HIT's BLEU score is higher than the median, then the HIT's color is dark. A light colored tick mark means the HIT's BLEU score is lower than the median. In Figure 1, the order of workers on y axis is based on the gold standard ranking of these workers. The top most worker ranked highest and the bottom most worker ranked lowest.

From Figure 1, we see that most workers' performance stays consistent as time passes. This is good, since it may enable us to predict workers' performance based on their early submission so that we can come to an early decision about whether to continue to hire them. If we rank the worker based solely on their first HITs' BLEU score, comparing all sentences in that HIT against the reference translation, then we get a surprisingly strong correlation with the true gold standard ranking of workers based on all of their submitted work. The Spearman Correlation is 0.86 when comparing this first-HIT ranking with gold standard ranking.

Expanding the experiment mentioned above, we rank workers using their first k HITs and calculate Spearman Correlation comparing against the gold standard ranking list (calculated over all HITs, instead of just the first k). The value of k could range from 1 to the max number of HITs submitted by any single worker. Since some workers only translated a small number of HITs, larger values of k will be greater than the total number of HITs that they provided. For those workers who translated less than k HITs in total, we use all of the HITs they submitted

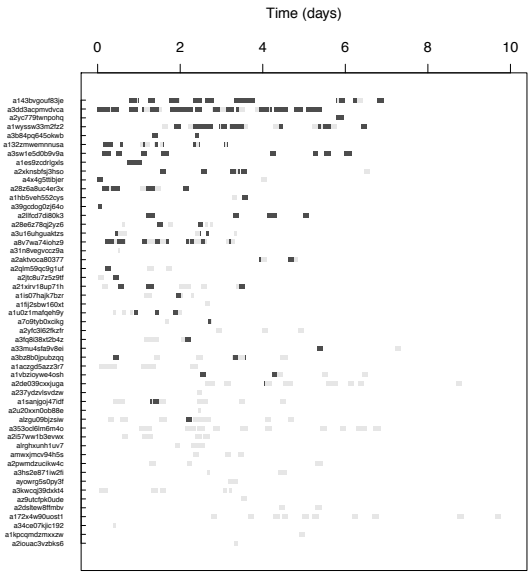

Figure 1: Workers' performance at across time.

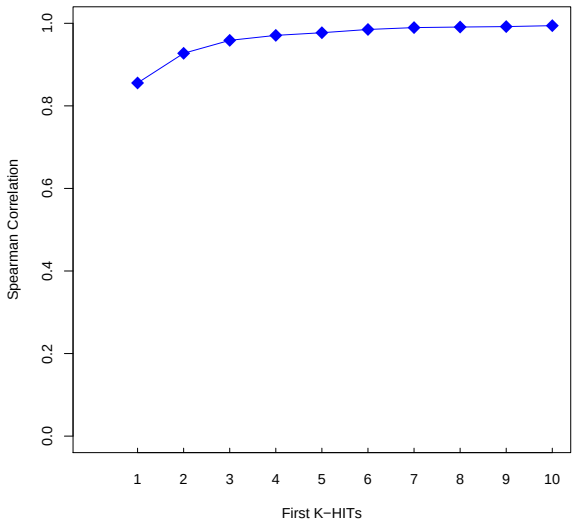

Figure 2: Spearman Correlation with the gold standard as we rank the workers based on their first k HITs

to rate them. Since the correlation converges to 1 after k is larger than 10, we plot the graph to show the increase in correlation for k from 1 to 10. Figure 2 shows details. The average number HITs each worker provided is 15.06, so we also plot Figure 3 to show the percentage of HITs we use to rank workers for each value of k.

4 Experiment

5 Related Work

For one of our approaches for lowering the costs of crowdsourcing, we train models to distinguish between acceptable and unacceptable translation candidates. To do so, we sweep a threshold of BLEU values. The threshold between acceptable and unacceptable translations is fuzzy so there exists some uncertainty in labeling each data sample. This is related to (Sheng et al., 2008)'s work on repeated labeling, which presents a way of solving the problems of uncertainty in labeling and selection of a threshold. In their work, they showed that for single-labeling examples, the labeling quality (the annotator's probability of producing a correct labeling) is critical to the model quality. The model prediction accuracy rises as the labeling quality increases. However, in reality, we cannot always get high-quality labeled data samples with relatively low

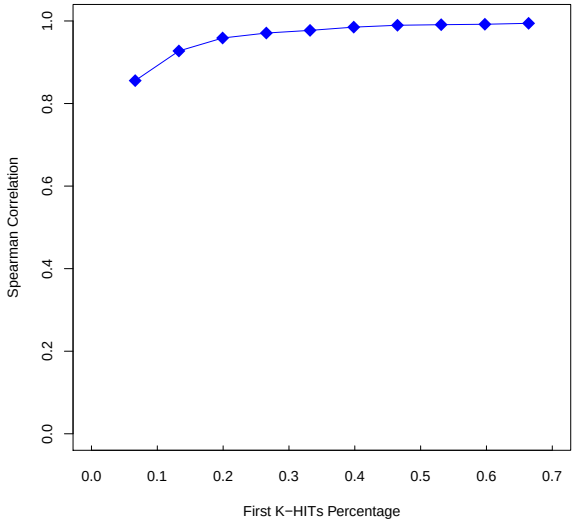

Figure 3: The percentage of worker data used to rank them as k increases

costs. To keep the model trained on noisy labeled data having a high accuracy in predicting, Sheng et al. (2008) proposed a framework for repeated-labeling that resolves the uncertainty in labeling via majority voting. The experimental results show that a model's predicting accuracy is improved even if labels in its training data are nosity and of imperfect quality. As long as the integrated quality (the probability of the integrated labeling being correct) is higher than 0.5, repeated labeling benefits model training. More closer the quality to 0.5, the more benefits obtained in model prediction.

A very important issue in natural language processing is data annotation. Hiring professional annotators is very expensive. As an alternative, collecting several annotation for each single data sample and pick the best label is more economical. In our work, we collected several translations for each source sentence and pick the best translation. Our work shares many goals in common with Passonneau and Carpenter (2013), who created a Bayesian model of annotation, which they applied to the problem of word sense annotation. Rather than hiring professional annotators, which is very expensive, they hire non-expert annotators on Mechanical Turk. They collected 20 to 25 word sense labels for each word. To decide which label to select for each word, and to compute the quality of the annotation, they proposed the probabilistic model using Bayes's rule. They calculated the product of the prior probability (the initial probability of being the observed label) and the conditional probability (the probability of being the observed label given the true label) and pick one label with the highest score. This sort of a probability estimate provides much more information about the corpus quality than previous methods, such as calculating inter-annotation agreement through Coehn's kappa score. Kappa measures the agreement coefficient among annotators in a chance-adjusted fashion. However, the method only reports how often annotators agree, but does not provide information about the quality of the corpus and the individual data sample.

Although Passonneau and Carpenter (2013) collect word sense labels, which are a small, enumerable set, and we collect translation (which could be thought of as a kind of label, albeit a very complex one), there is a strong commonality in the goals of their word and the goals of our work. Specifically, how can we use all the labels collected in order to select of the best label. And how can we rank the annotators themselves. For selecting the best label for word senses, majority voting is a direct and easy way to solve the problem, but the task is more complex for translation.

Passonneau and Carpenter (2013) also proposed an approach to detect and avoid spam workers. They measured the performance of worker by comparing worker's labels to the current majority labels and worker with bad performance would be blocked. However, this approach suffered from 2 shortcomings: (1) Sometimes majority labels may not reflect the ground truth label. (2) They didn't figure out how much data(HITs) is needed to evaluate a worker's performance. Although they could find the spam after the fact, it was a post-hoc analysis, so they had already paid for that worker and wasted the money. We attempt to identify poor workers as quickly as possible, in order to limit the amount of work that we solicit from them.

6 Discussion

7 Conclusion

Acknowledgments

Do not number the acknowledgment section. Do not include this section when submitting your paper for review.

References

Rebecca J Passonneau and Bob Carpenter. 2013. The benefits of a model of annotation. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 187–195. Citeseer.

Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM.

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.