Bucking the Trend: Large-Scale Cost-Focused Active Learning for Statistical Machine Translation

Michael Bloodgood
Human Language Technology
Center of Excellence
Johns Hopkins University
Baltimore, MD 21211
bloodgood@jhu.edu

Chris Callison-Burch
Center for Language and
Speech Processing
Johns Hopkins University
Baltimore, MD 21211
ccb@cs.jhu.edu

Abstract

We explore how to improve machine translation systems by adding more translation data in situations where we already have substantial resources. The main challenge is how to buck the trend of diminishing returns that is commonly encountered. We present an active learning-style data solicitation algorithm to meet this challenge. We test it, gathering annotations via Amazon Mechanical Turk, and find that we get an order of magnitude increase in performance rates of improvement.

1 Introduction

Figure 1 shows the learning curves for two state of the art statistical machine translation (SMT) systems for Urdu-English translation. Observe how the learning curves rise rapidly at first but then a trend of diminishing returns occurs: put simply, the curves flatten.

This paper investigates whether we can buck the trend of diminishing returns, and if so, how we can do it effectively. Active learning (AL) has been applied to SMT recently (Haffari et al., 2009; Haffari and Sarkar, 2009) but they were interested in starting with a tiny seed set of data, and they stopped their investigations after only adding a relatively tiny amount of data as depicted in Figure 1.

In contrast, we are interested in applying AL when a large amount of data already exists as is the case for many important lanuage pairs. We develop an AL algorithm that focuses on keeping annotation costs (measured by time in seconds) low. It succeeds in doing this by only soliciting translations for parts of sentences. We show that this gets a savings in human annotation time above and beyond what the reduction in # words annotated would have indicated by a factor of about three and speculate as to why.

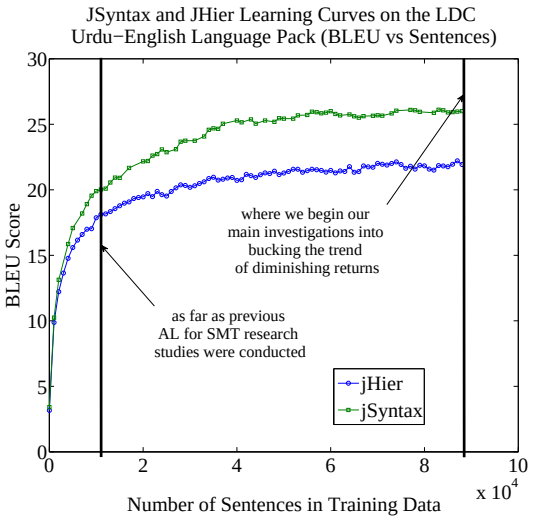

Figure 1: Syntax-based and Hierarchical Phrase-Based MT systems' learning curves on the LDC Urdu-English language pack. The x-axis measures the number of sentence pairs in the training data. The y-axis measures BLEU score. Note the diminishing returns as more data is added. Also note how relatively early on in the process previous studies were terminated. In contrast, the focus of our main experiments doesn't even begin until much higher performance has already been achieved with a period of diminishing returns firmly established.

We conduct experiments for Urdu-English translation, gathering annotations via Amazon Mechanical Turk (MTurk) and show that we can indeed buck the trend of diminishing returns, achieving an order of magnitude increase in the rate of improvement in performance.

Section 2 discusses related work; Section 3 discusses preliminary experiments that show the guiding principles behind the algorithm we use; Section 4 explains our method for soliciting new translation data; Section 5 presents our main results; and Section 6 concludes.

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 854–864,
Uppsala, Sweden, 11-16 July 2010.

2 Related Work

Active learning has been shown to be effective for improving NLP systems and reducing annotation burdens for a number of NLP tasks (see, e.g., (Hwa, 2000; Sassano, 2002; Bloodgood and Vijay-Shanker, 2008; Bloodgood and Vijay-Shanker, 2009b; Mairesse et al., 2010; Vickrey et al., 2010)). The current paper is most highly related to previous work falling into three main areas: use of AL when large corpora already exist; cost-focused AL; and AL for SMT.

In a sense, the work of Banko and Brill (2001) is closely related to ours. Though their focus is mainly on investigating the performance of learning methods on giant corpora many orders of magnitude larger than previously used, they do lay out how AL might be useful to apply to acquire data to augment a large set cheaply because they recognize the problem of diminishing returns that we discussed in Section 1.

The second area of work that is related to ours is previous work on AL that is cost-conscious. The vast majority of AL research has not focused on accurate cost accounting and a typical assumption is that each annotatable has equal annotation cost. An early exception in the AL for NLP field was the work of Hwa (2000), which makes a point of using # of brackets to measure cost for a syntactic analysis task instead of using # of sentences. Another relatively early work in our field along these lines was the work of Ngai and Yarowsky (2000), which measured actual times of annotation to compare the efficacy of rule writing versus annotation with AL for the task of BaseNP chunking. Osborne and Baldridge (2004) argued for the use of discriminant cost over unit cost for the task of Head Phrase Structure Grammar parse selection. King et al. (2004) design a robot that tests gene functions. The robot chooses which experiments to conduct by using AL and takes monetary costs (in pounds sterling) into account during AL selection and evaluation. Unlike our situation for SMT, their costs are all known beforehand because they are simply the cost of materials to conduct the experiments, which are already known to the robot. Hachey et al. (2005) showed that selectively sampled examples for an NER task took longer to annotate and had lower inter-annotator agreement. This work is related to ours because it shows that how examples are selected can impact the cost of annotation, an idea we turn around to use for our advantage when developing our data selection algorithm. Haertel et al. (2008) emphasize measuring costs carefully for AL for POS tagging. They develop a model based on a user study that can estimate the time required for POS annotating. Kapoor et al. (2007) assign costs for AL based on message length for a voicemail classification task. In contrast, we show for SMT that annotation times do not scale according to length in words and we show our method can achieve a speedup in annotation time above and beyond what the reduction in words would indicate. Tomanek and Hahn (2009) measure cost by # of tokens for an NER task. Their AL method only solicits labels for parts of sentences in the interest of reducing annotation effort. Along these lines, our method is similar in the respect that we also will only solicit annotation for parts of sentences, though we prefer to measure cost with time and we show that time doesn't track with token length for SMT.

Haffari et al. (2009), Haffari and Sarkar (2009), and Ambati et al. (2010) investigate AL for SMT. There are two major differences between our work and this previous work. One is that our intended use cases are very different. They deal with the more traditional AL setting of starting from an extremely small set of seed data. Also, by SMT standards, they only add a very tiny amount of data during AL. All their simulations top out at 10,000 sentences of labeled data and the models learned have relatively low translation quality compared to the state of the art.

On the other hand, in the current paper, we demonstrate how to apply AL in situations where we already have large corpora. Our goal is to buck the trend of diminishing returns and use AL to add data to build some of the highest-performing MT systems in the world while keeping annotation costs low. See Figure 1 from Section 1, which contrasts where (Haffari et al., 2009; Haffari and Sarkar, 2009) *stop* their investigations with where we *begin* our studies.

The other major difference is that (Haffari et al., 2009; Haffari and Sarkar, 2009) measure annotation cost by # of sentences. In contrast, we bring to light some potential drawbacks of this practice, showing it can lead to different conclusions than if other annotation cost metrics are used, such as time and money, which are the metrics that we use.

3 Simulation Experiments

Here we report on results of simulation experiments that help to illustrate and motivate the design decisions of the algorithm we present in Section 4. We use the Urdu-English language pack[1] from the Linguistic Data Consortium (LDC), which contains ≈ 88000 Urdu-English sentence translation pairs, amounting to ≈ 1.7 million Urdu words translated into English. All experiments in this paper evaluate on a genre-balanced split of the NIST2008 Urdu-English test set. In addition, the language pack contains an Urdu-English dictionary consisting of ≈ 114000 entries. In all the experiments, we use the dictionary at every iteration of training. This will make it harder for us to show our methods providing substantial gains since the dictionary will provide a higher base performance to begin with. However, it would be artificial to ignore dictionary resources when they exist.

We experiment with two translation models: hierarchical phrase-based translation (Chiang, 2007) and syntax augmented translation (Zollmann and Venugopal, 2006), both of which are implemented in the Joshua decoder (Li et al., 2009). We hereafter refer to these systems as jHier and jSyntax, respectively.

We will now present results of experiments with different methods for growing MT training data. The results are organized into three areas of investigations:

1. annotation costs;
2. managing uncertainty; and
3. how to automatically detect when to stop soliciting annotations from a pool of data.

3.1 Annotation Costs

We begin our cost investigations with four simple methods for growing MT training data: random, shortest, longest, and *VocabGrowth* sentence selection. The first three methods are self-explanatory. *VocabGrowth* (hereafter *VG*) selection is modeled after the best methods from previous work (Haffari et al., 2009; Haffari and Sarkar, 2009), which are based on preferring sentences that contain phrases that occur frequently in unlabeled data and infrequently in the so-far labeled data. Our *VG* method selects sentences for translation that contain n-grams (for n in {1,2,3,4}) that do not occur at all in our so-far labeled data. We call an n-gram "covered" if it occurs at least once in our so-far labeled data. *VG* has a preference for covering frequent n-grams before covering infrequent n-grams. The *VG* method is depicted in Figure 2.

Init:
Go through all available training data (labeled and unlabeled) and obtain frequency counts for every n-gram (n in $\{1, 2, 3, 4\}$) that occurs.
$sortedNGrams \leftarrow$ Sort n-grams by frequency in descending order.

Loop
until stopping criterion (see Section 3.3) is met
1. $trigger \leftarrow$ Go down $sortedNGrams$ list and find the first n-gram that isn't covered in the so far labeled training data.
2. $selectedSentence \leftarrow$ Find a sentence that contains $trigger$.
3. Remove $selectedSentence$ from unlabeled data and add it to labeled training data.

End Loop

Figure 2: The *VG* sentence selection algorithm

Figure 3 shows the learning curves for both jHier and jSyntax for *VG* selection and random selection. The y-axis measures BLEU score (Papineni et al., 2002),which is a fast automatic way of measuring translation quality that has been shown to correlate with human judgments and is perhaps the most widely used metric in the MT community. The x-axis measures the number of sentence translation pairs in the training data. The *VG* curves are cut off at the point at which the stopping criterion in Section 3.3 is met. From Figure 3 it might appear that *VG* selection is better than random selection, achieving higher-performing systems with fewer translations in the labeled data.

However, it is important to take care when measuring annotation costs (especially for relatively complicated tasks such as translation). Figure 4 shows the learning curves for the same systems and selection methods as in Figure 3 but now the x-axis measures the number of foreign words in the training data. The difference between *VG* and random selection now appears smaller.

For an extreme case, to illustrate the ramifica-

[1] LDC Catalog No.: LDC2006E110.

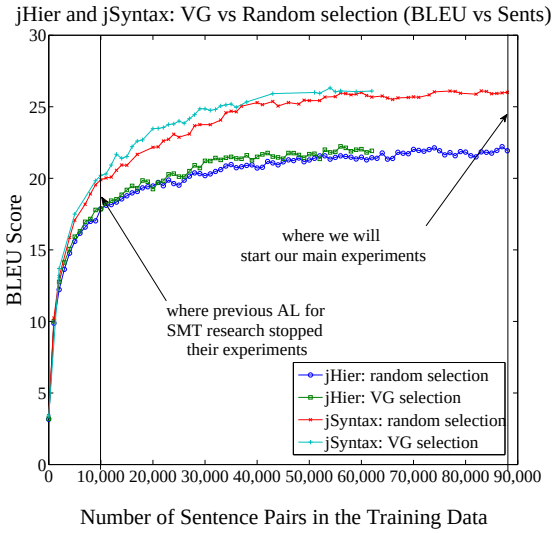

Figure 3: Random vs *VG* selection. The x-axis measures the number of sentence pairs in the training data. The y-axis measures BLEU score.

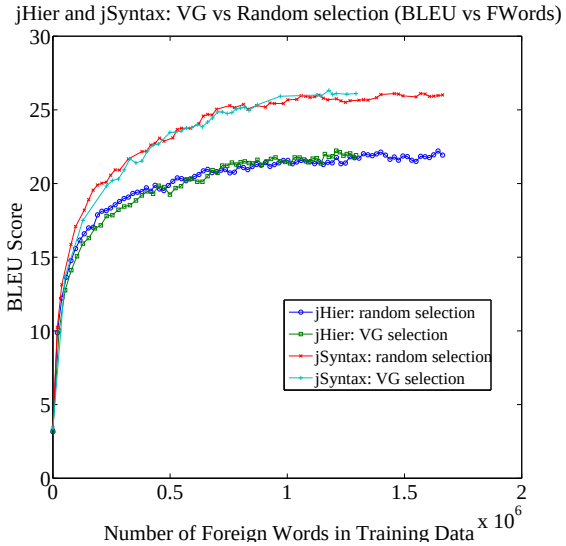

Figure 4: Random vs *VG* selection. The x-axis measures the number of foreign words in the training data. The y-axis measures BLEU score.

tions of measuring translation annotation cost by # of sentences versus # of words, consider Figures 5 and 6. They both show the same three selection methods but Figure 5 measures the x-axis by # of sentences and Figure 6 measures by # of words. In Figure 5, one would conclude that shortest is a far inferior selection method to longest but in Figure 6 one would conclude the opposite.

Measuring annotation time and cost in dollars are probably the most important measures of annotation cost. We can't measure these for the simulated experiments but we will use time (in seconds) and money (in US dollars) as cost measures in Section 5, which discusses our non-simulated AL experiments. If # sentences or # words track these other more relevant costs in predictable known relationships, then it would suffice to measure # sentences or # words instead. But it's clear that different sentences can have very different annotation time requirements according to how long and complicated they are so we will not use # sentences as an annotation cost any more. It is not as clear how # words tracks with annotation time. In Section 5 we will present evidence showing that time per word can vary considerably and also show a method for soliciting annotations that reduces time per word by nearly a factor of three.

As it is prudent to evaluate using accurate cost accounting, so it is also prudent to develop new AL algorithms that take costs carefully into account. Hence, reducing annotation time burdens instead of the # of sentences translated (which might be quite a different thing) will be a cornerstone of the algorithm we describe in Section 4.

3.2 Managing Uncertainty

One of the most successful of all AL methods developed to date is uncertainty sampling and it has been applied successfully many times (e.g.,(Lewis and Gale, 1994; Tong and Koller, 2002)). The intuition is clear: much can be learned (potentially) if there is great uncertainty. However, with MT being a relatively complicated task (compared with binary classification, for example), it might be the case that the uncertainty approach has to be re-considered. If words have never occurred in the training data, then uncertainty can be expected to be high. But we are concerned that if a sentence is translated for which (almost) no words have been seen in training yet, though uncertainty will be high (which is usually considered good for AL), the word alignments may be incorrect and then subsequent learning from that translation pair will be severely hampered.

We tested this hypothesis and Figure 7 shows empirical evidence that it is true. Along with *VG*, two other selection methods' learning curves are charted in Figure 7: mostNew, which prefers to select those sentences which have the largest # of unseen words in them; and moderateNew, which aims to prefer sentences that have a moderate # of unseen words, preferring sentences with $\approx$ ten

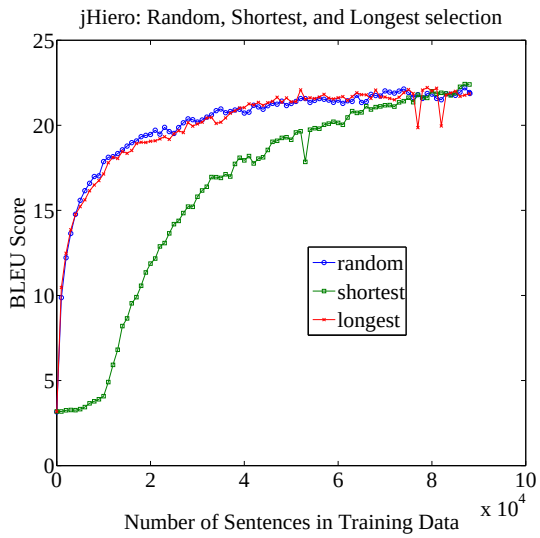

Figure 5: Random vs Shortest vs Longest selection. The x-axis measures the number of sentence pairs in the training data. The y-axis measures BLEU score.

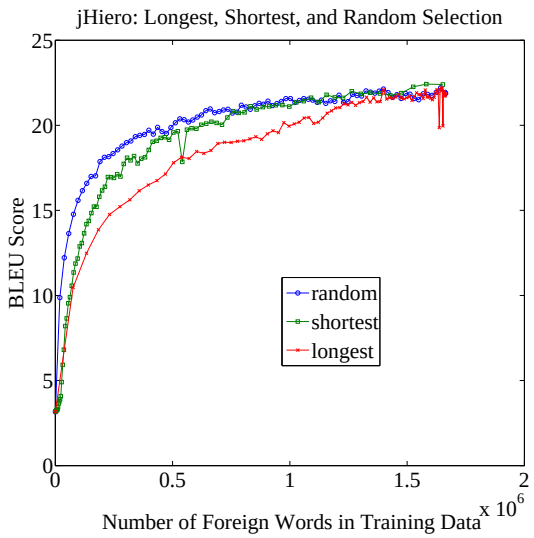

Figure 6: Random vs Shortest vs Longest selection. The x-axis measures the number of foreign words in the training data. The y-axis measures BLEU score.

unknown words in them. One can see that mostNew underperforms *VG*. This could have been due to *VG*'s frequency component, which mostNew doesn't have. But moderateNew also doesn't have a frequency preference so it is likely that mostNew winds up overwhelming the MT training system, word alignments are incorrect, and less is learned as a result. In light of this, the algorithm we develop in Section 4 will be designed to avoid this word alignment danger.

3.3 Automatic Stopping

The problem of automatically detecting when to stop AL is a substantial one, discussed at length in the literature (e.g., (Bloodgood and Vijay-Shanker, 2009a; Schohn and Cohn, 2000; Vlachos, 2008)). In our simulation, we stop *VG* once all n-grams (n in {1,2,3,4}) have been covered. Though simple, this stopping criterion seems to work well as can be seen by where the curve for *VG* is cut off in Figures 3 and 4. It stops after 1,293,093 words have been translated, with jHier's BLEU=21.92 and jSyntax's BLEU=26.10 at the stopping point. The ending BLEU scores (with the full corpus annotated) are 21.87 and 26.01 for jHier and jSyntax, respectively. So our stopping criterion saves 22.3% of the annotation (in terms of words) and actually achieves slightly higher BLEU scores than if all the data were used. Note: this "less is more" phenomenon has been commonly observed in AL settings (e.g., (Bloodgood and Vijay-Shanker, 2009a; Schohn and Cohn, 2000)).

4 Highlighted N-Gram Method

In this section we describe a method for soliciting human translations that we have applied successfully to improving translation quality in real (not simulated) conditions. We call the method the *Highlighted N-Gram* method, or *HNG*, for short. *HNG* solicits translations only for trigger n-grams and not for entire sentences. We provide sentential context, highlight the trigger n-gram that we want translated, and ask for a translation of just the highlighted trigger n-gram. *HNG* asks for translations for triggers in the same order that the triggers are encountered by the algorithm in Figure 2. A screenshot of our interface is depicted in Figure 8. The same stopping criterion is used as was used in the last section. When the stopping criterion becomes true, it is time to tap a new unlabeled pool of foreign text, if available.

Our motivations for soliciting translations for only parts of sentences are twofold, corresponding to two possible cases. Case one is that a translation model learned from the so-far labeled data will be able to translate most of the non-trigger words in the sentence correctly. Thus, by asking a human to translate only the trigger words, we avoid wasting human translation effort. (We will show in

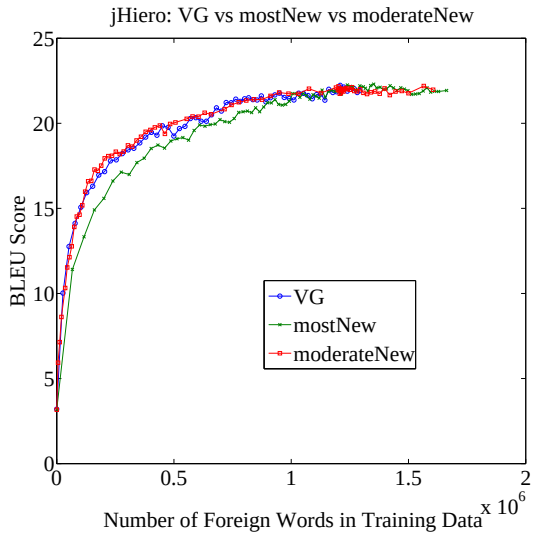

Figure 7: *VG* vs MostNew vs ModerateNew selection. The x-axis measures the number of sentence pairs in the training data. The y-axis measures BLEU score.

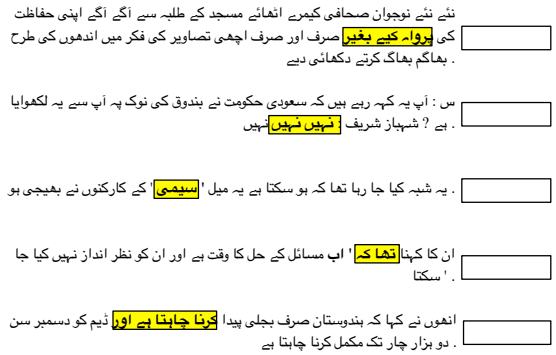

Figure 8: Screenshot of the interface we used for soliciting translations for triggers.

the next section that we even get a much larger speedup above and beyond what the reduction in number of translated words would give us.) Case two is that a translation model learned from the so-far labeled data will (in addition to not being able to translate the trigger words correctly) also not be able to translate most of the non-trigger words correctly. One might think then that this would be a great sentence to have translated because the machine can potentially learn a lot from the translation. Indeed, one of the overarching themes of AL research is to query examples where uncertainty is greatest. But, as we showed evidence for in the last section, for the case of SMT, too much uncertainty could in a sense overwhelm the machine and it might be better to provide new training data in a more gradual manner. A sentence with large #s of unseen words is likely to get word-aligned incorrectly and then learning from that translation could be hampered. By asking for a translation of only the trigger words, we expect to be able to circumvent this problem in large part.

The next section presents the results of experiments that show that the *HNG* algorithm is indeed practically effective. Also, the next section analyzes results regarding various aspects of *HNG*'s behavior in more depth.

5 Experiments and Discussion

5.1 General Setup

We set out to see whether we could use the *HNG* method to achieve translation quality improvements by gathering additional translations to add to the training data of the entire LDC language pack, including its dictionary. In particular, we wanted to see if we could achieve translation improvements on top of already state-of-the-art performing systems trained already on the *entire* LDC corpus. Note that at the outset this is an ambitious endeavor (recall the flattening of the curves in Figure 1 from Section 1).

Snow et al. (2008) explored the use of the Amazon Mechanical Turk (MTurk) web service for gathering annotations for a variety of natural language processing tasks and recently MTurk has been shown to be a quick, cost-effective way to gather Urdu-English translations (Bloodgood and Callison-Burch, 2010). We used the MTurk web service to gather our annotations. Specifically, we first crawled a large set of BBC articles on the internet in Urdu and used this as our unlabeled pool from which to gather annotations. We applied the *HNG* method from Section 4 to determine what to post on MTurk for workers to translate.[2] We gathered 20,580 n-gram translations for which we paid $0.01 USD per translation, giving us a total cost of $205.80 USD. We also gathered 1632 randomly chosen Urdu sentence translations as a control set, for which we paid $0.10 USD per sentence translation.[3]

[2]For practical reasons we restricted ourselves to not considering sentences that were longer than 60 Urdu words, however.

[3]The prices we paid were not market-driven. We just chose prices we thought were reasonable. In hindsight, given how much quicker the phrase translations are for people we could have had a greater disparity in price.

5.2 Accounting for Translation Time

MTurk returns with each assignment the "WorkTimeInSeconds." This is the amount of time between when a worker accepts an assignment and when the worker submits the completed assignment. We use this value to estimate annotation times.[4]

Figure 9 shows *HNG* collection versus random collection from MTurk. The x-axis measures the number of seconds of annotation time. Note that *HNG* is more effective. A result that may be particularly interesting is that *HNG* results in a time speedup by more than just the reduction in translated words would indicate. The average time to translate a word of Urdu with the sentence postings to MTurk was 32.92 seconds. The average time to translate a word with the *HNG* postings to MTurk was 11.98 seconds. This is nearly three times faster. Figure 10 shows the distribution of speeds (in seconds per word) for *HNG* postings versus complete sentence postings. Note that the *HNG* postings consistently result in faster translation speeds than the sentence postings[5].

We hypothesize that this speedup comes about because when translating a full sentence, there's the time required to examine each word and translate them in some sense (even if not one-to-one) and then there is an extra significant overhead time to put it all together and synthesize into a larger sentence translation. The factor of three speedup is evidence that this overhead is significant effort compared to just quickly translating short n-grams from a sentence. This speedup is an additional benefit of the *HNG* approach.

5.3 Bucking the Trend

We gathered translations for ≈ 54,500 Urdu words via the use of *HNG* on MTurk. This is a relatively small amount, ≈ 3% of the LDC corpus. Figure 11 shows the performance when we add this training data to the LDC corpus. The rectangle around the last 700,000 words of the LDC data is wide and short (it has a height of 0.9 BLEU points and a width of 700,000 words) but the rectangle around the newly added translations is narrow and tall (a height of 1 BLEU point and a width of 54,500 words). Visually, it appears we are succeeding in bucking the trend of diminishing returns. We further confirmed this by running a least-squares linear regression on the points of the last 700,000 words annotated in the LDC data and also for the points in the new data that we acquired via MTurk for $205.80 USD. We find that the slope fit to our new data is 6.6245E-06 BLEU points per Urdu word, or 6.6245 BLEU points for a million Urdu words. The slope fit to the LDC data is only 7.4957E-07 BLEU points per word, or only 0.74957 BLEU points for a million words. This is already an order of magnitude difference that would make the difference between it being worth adding more data and not being worth it; and this is leaving aside the added time speedup that our method enjoys.

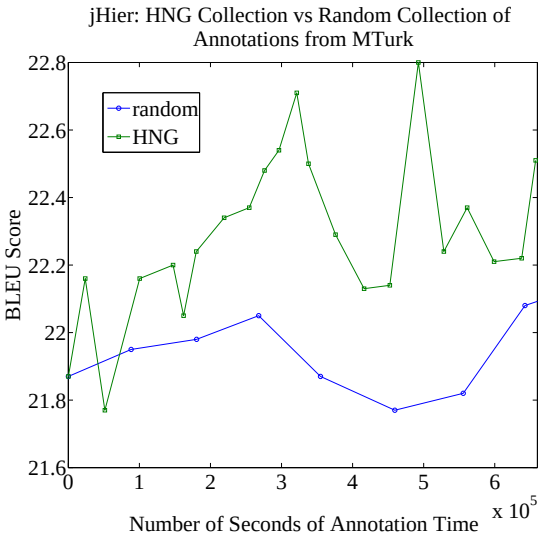

Figure 9: *HNG* vs Random collection of new data via MTurk. y-axis measures BLEU. x-axis measures annotation time in seconds.

Still, we wondered why we could not have raised BLEU scores even faster. The main hurdle seems to be one of coverage. Of the 20,580 n-grams we collected, only 571 (i.e., 2.77%) of them ever even occur in the test set.

5.4 Beyond BLEU Scores

BLEU is an imperfect metric (Callison-Burch et al., 2006). One reason is that it rates all ngram

[4]It's imperfect because of network delays and if a person is multitasking or pausing between their accept and submit times. Nonetheless, the times ought to be better estimates as they are taken over larger samples.

[5]The average speed for the *HNG* postings seems to be slower than the histogram indicates. This is because there were a few extremely slow outlier speeds for a handful of *HNG* postings. These are almost certainly not cases when the turker is working continuously on the task and so the average speed we computed for the *HNG* postings might be slower than the actual speed and hence the true speedup may even be faster than indicated by the difference between the average speeds we reported.

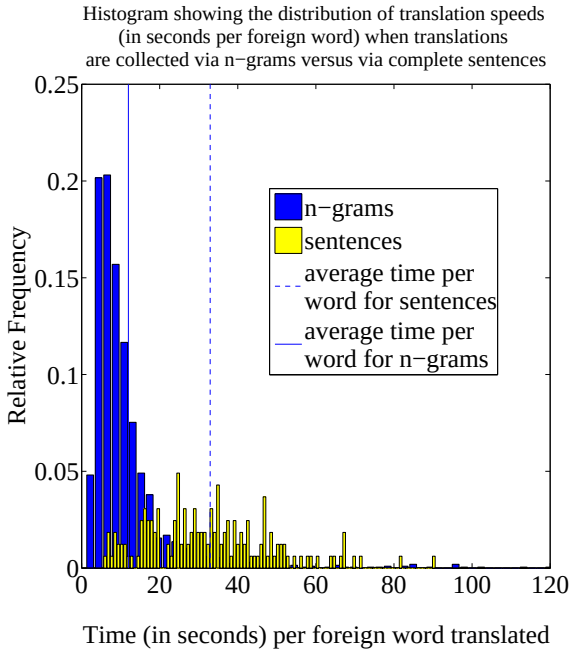

Figure 10: Distribution of translation speeds (in seconds per word) for *HNG* postings versus complete sentence postings. The y-axis measures relative frequency. The x-axis measures translation speed in seconds per word (so farther to the left is faster).

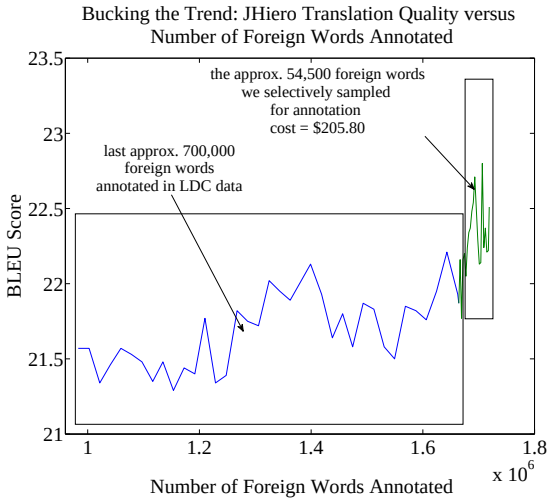

Figure 11: Bucking the trend: performance of *HNG*-selected additional data from BBC web crawl data annotated via Amazon Mechanical Turk. y-axis measures BLEU. x-axis measures number of words annotated.

- Learned phrase: "ناگالینڈ" means "nagaland"
- The "Before System" translation:
  - according to police the number of hundreds of ناگالینڈarmed tribesmen in the گلیکیof assam and fell سیبساset fire in three villages .
- The "After System" translation:
  - according to police the number of hundreds of nagaland armed tribesmen in the گلیکیof assam and fell سیبساset fire in three villages .
- Reference Translation:
  - according to the police , hundreds of armed nagaland tribesmen set on fire three villages in galleci and sebsagarh areas of assam .

Figure 12: Example of strategy working.

mismatches equally although some are much more important than others. Another reason is it's not intuitive what a gain of x BLEU points means in practice. Here we show some concrete example translations to show the types of improvements we're achieving and also some examples which suggest improvements we can make to our AL selection algorithm in the future. Figure 12 shows a prototypical example of our system working.

Figure 13 shows an example where the strategy is working partially but not as well as it might. The Urdu phrase was translated by turkers as "gowned veil". However, since the word aligner just aligns the word to "gowned", we only see "gowned" in our output. This prompts a number of discussion points. First, the 'after system' has better translations but they're not rewarded by BLEU scores because the references use the words 'burqah' or just 'veil' without 'gowned'. Second, we hypothesize that we may be able to see improvements by overriding the automatic alignment software whenever we obtain a many-to-one or one-to-many (in terms of words) translation for one of our trigger phrases. In such cases, we'd like to make sure that every word on the 'many' side is aligned to the single word on the 'one' side. For example, we would force both 'gowned' and 'veil' to be aligned to the single Urdu word instead of allowing the automatic aligner to only align 'gowned'.

Figure 14 shows an example where our "before" system already got the translation correct without the need for the additional phrase translation. This is because though the "before" system had never seen the Urdu expression for "12 May", it had seen the Urdu words for "12" and "May" in isolation and was able to successfully compose them. An area of future work is to use the "before" system to determine such cases automatically and avoid asking humans to provide translations in such cases.

- Learned phrase: "برقعے" means "gowned veil"
- The "Before System" translation:
  - ' in برقعےmaulana in برقعےgeneral is not islam in ! برقعے
- The "After System" translation:
  - ' is in gowned , maulana gowned , in general is not islam in gowned !
- Reference translations:
  - in burqah , there is the maulana . in burqah , there is the general . in burqah , there is no islam .
  - molana is under veil , and the general is under veil , but islam is not under veil .

Figure 13: Example showing where we can improve our selection strategy.

- Example: "بارہ مئی" means "12 may"
  - The "Before System" translation:
    - all party conference in speeches during the meeting two days of karachi on 12 may in connection with the incidents , was strongly criticized the mqm .
  - The "After System" translation:
    - during the meeting two days of all party conference held in karachi on 12 may in the speeches about the incidents were strongly criticized the mqm .

Figure 14: Example showing where we can improve our selection strategy.

6 Conclusions and Future Work

We succeeded in bucking the trend of diminishing returns and improving translation quality while keeping annotation costs low. In future work we would like to apply these ideas to domain adaptation (say, general-purpose MT system to work for scientific domain such as chemistry). Also, we would like to test with more languages, increase the amount of data we can gather, and investigate stopping criteria further. Also, we would like to investigate increasing the efficiency of the selection algorithm by addressing issues such as the one raised by the 12 May example presented earlier.

Acknowledgements

This work was supported by the Johns Hopkins University Human Language Technology Center of Excellence. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor.

References

Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. 2010. Active learning and crowd-sourcing for machine translation. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 26–33, Toulouse, France, July. Association for Computational Linguistics.

Michael Bloodgood and Chris Callison-Burch. 2010. Using mechanical turk to build machine translation evaluation sets. In *Proceedings of the Workshop on Creating Speech and Language Data With Amazon's Mechanical Turk*, Los Angeles, California, June. Association for Computational Linguistics.

Michael Bloodgood and K Vijay-Shanker. 2008. An approach to reducing annotation costs for bionlp. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 104–105, Columbus, Ohio, June. Association for Computational Linguistics.

Michael Bloodgood and K Vijay-Shanker. 2009a. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 39–47, Boulder, Colorado, June. Association for Computational Linguistics.

Michael Bloodgood and K Vijay-Shanker. 2009b. Taking into account the differences between actively and passively acquired data: The case of active learning with support vector machines for imbalanced datasets. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 137–140, Boulder, Colorado, June. Association for Computational Linguistics.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, Trento, Italy.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Ben Hachey, Beatrice Alex, and Markus Becker. 2005. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 144–151, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Robbie Haertel, Eric Ringger, Kevin Seppi, James Carroll, and Peter McClanahan. 2008. Assessing the

costs of sampling methods in active learning for annotation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 65–68, Columbus, Ohio, June. Association for Computational Linguistics.

Gholamreza Haffari and Anoop Sarkar. 2009. Active learning for multilingual statistical machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 181–189, Suntec, Singapore, August. Association for Computational Linguistics.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 415–423, Boulder, Colorado, June. Association for Computational Linguistics.

Rebecca Hwa. 2000. Sample selection for statistical grammar induction. In Hinrich Schütze and Keh-Yih Su, editors, *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing*, pages 45–53. Association for Computational Linguistics, Somerset, New Jersey.

Ashish Kapoor, Eric Horvitz, and Sumit Basu. 2007. Selective supervision: Guiding supervised learning with decision-theoretic active learning. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 877–882.

Ross D. King, Kenneth E. Whelan, Ffion M. Jones, Philip G. K. Reiser, Christopher H. Bryant, Stephen H. Muggleton, Douglas B. Kell, and Stephen G. Oliver. 2004. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427:247–252, 15 January.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, New York, NY, USA. Springer-Verlag New York, Inc.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. 2009. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139, Athens, Greece, March. Association for Computational Linguistics.

Francois Mairesse, Milica Gasic, Filip Jurcicek, Simon Keizer, Jorge Prombonas, Blaise Thomson, Kai Yu, and Steve Young. 2010. Phrase-based statistical language generation using graphical models and active learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden, July. Association for Computational Linguistics.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Miles Osborne and Jason Baldridge. 2004. Ensemble-based active learning for parse selection. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 89–96, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Manabu Sassano. 2002. An empirical study of active learning with support vector machines for japanese word segmentation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 505–512, Morristown, NJ, USA. Association for Computational Linguistics.

Greg Schohn and David Cohn. 2000. Less is more: Active learning with support vector machines. In *Proc. 17th International Conf. on Machine Learning*, pages 839–846. Morgan Kaufmann, San Francisco, CA.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, October. Association for Computational Linguistics.

Katrin Tomanek and Udo Hahn. 2009. Semi-supervised active learning for sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1039–1047, Suntec, Singapore, August. Association for Computational Linguistics.

Simon Tong and Daphne Koller. 2002. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research (JMLR)*, 2:45–66.

David Vickrey, Oscar Kipersztok, and Daphne Koller. 2010. An active learning approach to finding related terms. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden, July. Association for Computational Linguistics.

Andreas Vlachos. 2008. A stopping criterion for active learning. *Computer Speech and Language*, 22(3):295–312.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of the NAACL-2006 Workshop on Statistical Machine Translation (WMT06)*, New York, New York.