A Systematic Exploration of Diversity in Machine Translation

Kevin Gimpel[*] **Dhruv Batra**[†] **Chris Dyer**[‡] **Gregory Shakhnarovich**[*]
[*]Toyota Technological Institute at Chicago, Chicago, IL 60637, USA
[†]Virginia Tech, Blacksburg, VA 24061, USA
[‡]Carnegie Mellon University, Pittsburgh, PA 15213, USA
Corresponding author: kgimpel@ttic.edu

Abstract

This paper addresses the problem of producing a diverse set of plausible translations. We present a simple procedure that can be used with any statistical machine translation (MT) system. We explore three ways of using diverse translations: (1) system combination, (2) discriminative reranking with rich features, and (3) a novel post-editing scenario in which multiple translations are presented to users. We find that diversity can improve performance on these tasks, especially for sentences that are difficult for MT.

1 Introduction

From the perspective of user interaction, the ideal machine translator is an agent that reads documents in one language and produces accurate, high quality translations in another. This interaction ideal has been implicit in machine translation (MT) research since the field's inception. It is the way we interact with commercial MT services (such as Google Translate and Microsoft Translator), and the way MT systems are evaluated (Bojar et al., 2013). Unfortunately, when a real, imperfect MT system makes an error, the user is left trying to guess what the original sentence means.

Multiple Hypotheses. In contrast, when we look at the way other *computer systems* consume output from MT systems (or similarly unreliable tools), we see a different pattern. In a pipeline setting it is commonplace to propagate not just a *single*-best output but the M-best hypotheses (Venugopal et al., 2008). Multiple solutions are also used for reranking (Collins, 2000; Shen and Joshi, 2003; Collins and Koo, 2005; Charniak and Johnson, 2005), tuning (Och, 2003), minimum Bayes risk decoding (Kumar and Byrne, 2004), and system combination (Rosti et al., 2007). When dealing with error-prone systems, knowing about alternatives has benefits over relying on only a single output (Finkel et al., 2006; Dyer, 2010).

Need for Diversity. Unfortunately, M-best lists are a poor surrogate for structured output spaces (Finkel et al., 2006; Huang, 2008). In MT, for example, many translations on M-best lists are extremely similar, often differing only by a single punctuation mark or minor morphological variation. Recent work has explored reasoning about sets using packed representations such as lattices and hypergraphs (Macherey et al., 2008; Tromble et al., 2008; Kumar et al., 2009), or sampling translations proportional to their probability (Chatterjee and Cancedda, 2010). We argue that the implicit goal behind these techniques is to better explore the output space by introducing diversity into the surrogate set.

Overview and Contributions. In this work, we elevate diversity to a first-class status and directly address the problem of generating a set of **diverse, plausible translations**. We use the recently proposed technique of Batra et al. (2012), which produces diverse M-best solutions from a probabilistic model using a generic **dissimilarity** function $\Delta(\cdot, \cdot)$ that specifies how two solutions differ. Our first contribution is a family of dissimilarity functions for MT that admit simple algorithms for generating diverse translations. Other contributions are empirical: we show that diverse translations can lead to improvements for system combination and discriminative reranking. We also perform a novel human

post-editing evaluation in order to measure whether diverse translations can help users make sense of noisy MT output. We find that diverse translations can help post-editors produce better outputs for sentences that are the most difficult for MT. While we focus on machine translation in this paper, we note that our approach is applicable to other structure prediction problems in NLP.

2 Preliminaries and Notation

Let $\mathcal{X}$ denote the set of all strings in a source language. For an $x \in \mathcal{X}$, let $\mathcal{Y}_x$ denote the set of its possible translations $\mathbf{y}$ in the target language. MT models typically include a latent variable that captures the derivational structure of the translation process. Regardless of its specific form, we refer to this variable as a **derivation** $\mathbf{h} \in \mathcal{H}_x$, where $\mathcal{H}_x$ is the set of possible values of $\mathbf{h}$ for x. Derivations are coupled with translations and we define $\mathcal{T}_x \subseteq \mathcal{Y}_x \times \mathcal{H}_x$ as the set of possible $\langle \mathbf{y}, \mathbf{h} \rangle$ pairs for x.

We use a linear model with a parameter vector $\mathbf{w}$ and a vector $\boldsymbol{\phi}(x, \mathbf{y}, \mathbf{h})$ of feature functions on x, $\mathbf{y}$, and $\mathbf{h}$ (Och and Ney, 2002). The translation of x is selected using a simple decision rule:

$$\langle \hat{\mathbf{y}}, \hat{\mathbf{h}} \rangle = \underset{\langle \mathbf{y}, \mathbf{h} \rangle \in \mathcal{T}_x}{\operatorname{argmax}} \ \mathbf{w}^\intercal \boldsymbol{\phi}(x, \mathbf{y}, \mathbf{h}) \qquad (1)$$

where we also maximize over the latent variable $\mathbf{h}$ for efficiency. Translation models differ in the form of $\mathcal{T}_x$ and the choice of the feature functions $\boldsymbol{\phi}$. In this paper we focus on phrase-based (Koehn et al., 2003) and hierarchical phrase-based (Chiang, 2007) models, which include several bilingual and monolingual features, including n-gram language models.

3 Diversity in Machine Translation

We now address the task of producing a set of diverse high-scoring translations.

3.1 Generating Diverse Translations

We use a recently proposed technique (Batra et al., 2012) that constructs diverse lists via a greedy iterative procedure as follows. Let $\mathbf{y}^1$ be the model-best translation (Eq. 1). On the m-th iteration, the m-th best (diverse) translation is obtained as $\langle \mathbf{y}^m, \mathbf{h}^m \rangle =$

$$\underset{\langle \mathbf{y}, \mathbf{h} \rangle \in \mathcal{T}_x}{\operatorname{argmax}} \ \mathbf{w}^\intercal \boldsymbol{\phi}(x, \mathbf{y}, \mathbf{h}) + \sum_{j=1}^{m-1} \lambda_j \Delta(\mathbf{y}^j, \mathbf{y}) \qquad (2)$$

where Δ is a dissimilarity function and λ_j is the weight placed on dissimilarity to previous translation j relative to the model score. Intuitively, we seek a translation that is highly-scoring under the model while being different (as measured by Δ) from all previous translations. The λ parameters determine the trade-off between model score and diversity. We refer to Eq. (2) as **dissimilarity-augmented** decoding.

The objective in Eq. (2) is a Lagrangian relaxation for an intractable constrained objective specifying a minimum dissimilarity Δ_{min} between translations in the list, i.e., $\Delta(\mathbf{y}^j, \mathbf{y}) \geq \Delta_{min}$ (Batra et al., 2012). Instead of setting the dissimilarity threshold Δ_{min}, we set the weights λ_j. While the formulation allows for a different λ_j for each previous solution j, we simply use a single $\lambda = \lambda_j$ for all j. This was also done in the experiments in (Batra et al., 2012).

Note that if the dissimilarity function factors across the parts of the output variables $\langle \mathbf{y}, \mathbf{h} \rangle$ in the same way as the features $\boldsymbol{\phi}$, then *the same decoding algorithm* can be used as for Eq. (1). We discuss design choices for Δ next.

3.2 Dissimilarity Functions for MT

When designing a dissimilarity function $\Delta(\cdot, \cdot)$ for MT, we want to consider variation both in individual word choice and longer-range sentence structure. We also want a function that can be easily incorporated into extant statistical MT systems. We propose a dissimilarity function that simply counts the number of times any n-gram is present in both translations, then negates. Letting $q = n - 1$:

$$\Delta_n(\mathbf{y}, \mathbf{y}') = -\sum_{i=1}^{|\mathbf{y}|-q} \sum_{j=1}^{|\mathbf{y}'|-q} [\![\mathbf{y}_{i:i+q} = \mathbf{y}'_{j:j+q}]\!] \qquad (3)$$

where $[\![\cdot]\!]$ is the Iverson bracket (1 if input condition is true, 0 otherwise) and $\mathbf{y}_{i:j}$ is the subsequence of $\mathbf{y}$ from word i to word j (inclusive).

Importantly, Eq. (2) can be solved with no change to the decoding algorithm. The dissimilarity terms can simply be incorporated as an additional language model in ARPA format that sets the log-probability to the negated count for each n-gram in previous diverse translations, and sets to zero all other n-grams' log-probabilities and back-off weights.

The advantage of this dissimilarity function is its simplicity. It can be easily used with any translation system that uses n-gram language models without any change to the decoder. Indeed, we use both phrase-based and hierarchical phrase-based models in our experiments below.

4 Related Work

MT researchers have recently started to consider diversity in the context of system combination (Macherey and Och, 2007). Most closely-related is work by Devlin and Matsoukas (2012), who proposed a way to generate diverse translations by varying particular "traits," such as translation length, number of rules applied, etc. Their approach can be viewed as solving Eq. (2) with a richer dissimilarity function that requires a special-purpose decoding algorithm. We chose our n-gram dissimilarity function due to its simplicity and applicability to most MT systems without requiring any change to decoders.

Among other work, Xiao et al. (2013) used bagging and boosting to get diverse system outputs for system combination and Cer et al. (2013) used multiple identical systems trained jointly with an objective function that encourages the systems to generate complementary translations.

There is also similarity between our approach and minimum Bayes risk decoding (Kumar and Byrne, 2004), variational decoding (Li et al., 2009), and other "consensus" decoding algorithms (DeNero et al., 2009). These all seek a single translation that is most similar on average to the model's preferred translations. In this way, they try to capture the model's range of beliefs in a single translation. We instead seek a *set* of translations that, when considered as a whole, similarly express the full range of the model's beliefs about plausible translations for the input.

Also related is work on determinantal point processes (DPPs; Kulesza and Taskar, 2010), an elegant probabilistic model over sets of items that naturally prefers diverse sets. DPPs have been applied to summarization (Kulesza and Taskar, 2011) and discovery of topical threads in document collections (Gillenwater et al., 2012). Unfortunately, in the *structured* setting, DPPs make severely restrictive assumptions on the scoring function, while our framework does not.

5 Experimental Setup

We now embark on an extensive empirical evaluation of the framework presented above. We begin by analyzing our diverse sets of translations, showing how they differ from standard M-best lists (Section 6), followed by three tasks that illustrate how diversity can be exploited to improve translation quality: system combination (Section 7), discriminative reranking (Section 8), and a novel human post-editing task (Section 9). In the remainder of this section, we describe details of our experimental setup.

5.1 Language Pairs and Datasets

We use three language pairs: Arabic-to-English (AR→EN), Chinese-to-English (ZH→EN), and German-to-English (DE→EN). For AR→EN and DE→EN, we used a phrase-based model (Koehn et al., 2003) and for ZH→EN we used a hierarchical phrase-based model (Chiang, 2007).

Each language pair has two tuning and one test set: TUNE1 is used for tuning the baseline systems with minimum error rate training (MERT; Och, 2003), TUNE2 is used for training system combiners and rerankers, and TEST is used for evaluation. There are four references for AR→EN and ZH→EN and one for DE→EN.

For AR→EN, we used data provided by the LDC for the NIST evaluations, which includes 3.3M sentences of UN data and 982K sentences from other (mostly news) sources. Arabic text was preprocessed using an HMM segmenter that splits attached prepositional phrases, personal pronouns, and the future marker (Lee et al., 2003). The common stylistic sentence-initial *w+* (*and*) clitic was removed. The resulting corpus contained 130M Arabic tokens and 130M English tokens. We used the NIST MT06 test set as TUNE1, a 764-sentence subset of MT05 as TUNE2, and MT08 as TEST.

For ZH→EN, we used 303k sentence pairs from the FBIS corpus (LDC2003E14). We segmented the Chinese data using the Stanford Chinese segmenter (Chang et al., 2008) in "CTB" mode, giving us 7.9M Chinese tokens and 9.4M English tokens. We used the NIST MT02 test set as TUNE1, MT05

as TUNE2, and MT03 as TEST.

For DE→EN, we used data released for the WMT2011 shared task (Callison-Burch et al., 2011). German compound words were split using a CRF segmenter (Dyer, 2009). We used the WMT2010 test set as TUNE1, the 2009 test set as TUNE2, and the 2011 test set as TEST.

5.2 Baseline Systems

We used the Moses MT toolkit (Koehn et al., 2007; Hoang et al., 2009) with default settings and features for both phrase-based and hierarchical systems. Word alignment was done using GIZA++ (Och and Ney, 2003) in both directions, with the `grow-diag-final-and` heuristic used to symmetrize the alignments and a max phrase length of 7 used for phrase extraction.

Language models used the target side of the parallel corpus in each case augmented with 24.8M lines (601M tokens) of randomly-selected sentences from the Gigaword v4 corpus (excluding the NY Times and LA Times). We used 5-gram models, estimated using the SRI Language Modeling toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing (Chen and Goodman, 1998). The minimum count cut-off for unigrams, bigrams, and trigrams was 1 and the cut-off for 4-grams and 5-grams was 3. Language model inference used KenLM (Heafield, 2011).

Uncased IBM BLEU was used for evaluation (Papineni et al., 2002). MERT was used to train the feature weights for the baseline systems on TUNE1. We used the learned parameters to generate M-best and diverse lists for TUNE2 and TEST to use for subsequent experiments.

5.3 Diverse List Generation

Generating diverse translations depends on two hyperparameters: the n-gram order used by the dissimilarity function Δ_n (§3.2) and the λ_j weights on the dissimilarity terms in Eq. (2). Though our framework permits different λ_j for each j, we use a single λ value for simplicity, as was also done in (Batra et al., 2012). The values of n and λ were tuned on a 200 sentence subset of TUNE1 separately for each language pair (which we call TUNE_{200}), so as to maximize the oracle BLEU score of the diverse lists.[1] We considered n values in $\{2, 3, \ldots, 9\}$ and λ values in $\{0.005, 0.01, 0.05, 0.1\}$. We give details on optimal values for these hyperparameters when discussing particular tasks below.

	AR→EN	ZH→EN	DE→EN
1 best	50.1	36.9	21.8
20 best	54.0	40.3	24.7
200 best	57.5	43.8	27.7
1000 best	59.8	46.4	29.8
unique 20 best	56.6	44.1	26.7
unique 200 best	59.6	46.4	29.5
20 diverse	58.5	46.4	28.6
20 div × 10 best	61.3	48.7	30.3
20 div × 50 best	**63.2**	**50.6**	**31.6**

Table 1: Oracle BLEU scores on TEST for various sizes of M-best and diverse lists. Unique lists were obtained from 1,000-best lists and therefore may not contain the target number of unique translations for all sentences.

Though simple, our approach is computationally expensive as M grows because it requires decoding M times for each sentence. So, we assume $M \leq 20$. But we also extract an N-best list for each of the M diverse translations.[2] Many MT decoders, including the phrase-based and hierarchical implementations in Moses, permit efficient extraction of N-best lists, so we exploit this to obtain larger lists that still exhibit diversity. But we note that these N-best lists for each diverse solution are not in themselves diverse; with more computational power or more efficient algorithms (Devlin and Matsoukas, 2012) we could potentially generate larger, more diverse lists.

6 Analysis of Diverse Lists

We now characterize our diverse lists by comparing them to M-best lists. Table 1 shows oracle BLEU scores on TEST for M-best lists, unique M-best lists, and diverse lists of several sizes. To get unique lists, we first generated 1000-best lists, then retained only the highest-scoring derivation for each unique translation. When comparing M-best and diverse lists of comparable size, the diverse lists al-

[1] Since BLEU does not decompose additively across segments, we chose translations for individual sentences that maximized BLEU+1 (Lin and Och, 2004), then computed "oracle" corpus BLEU of these translations.

[2] We did not consider n-grams from previous N-best lists when computing the dissimilarity function, but only those from the previous *diverse* translations.

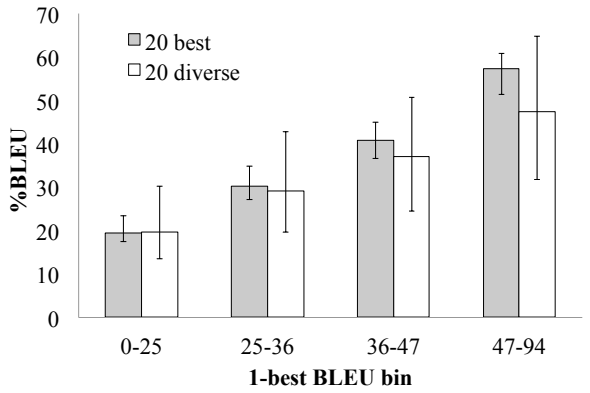

Figure 1: Median, min, and max BLEU+1 of 20-best and 20-diverse lists for the ZH→EN test set, divided into quartiles according to the BLEU+1 score of the 1-best translation, and averaged across sentences in each quartile. Heights of the bars show median and "error bars" indicate max and min.

ways have higher oracle BLEU. The differences are largest when comparing 20-best lists and 20-diverse lists, where they range from 4 to 6 BLEU points.

When generating these diverse lists, we used the n and λ values that were tuned for each language pair to maximize oracle BLEU on TUNE_{200} for the "20 div $\times$ 50 best" configuration. The optimal values of n were 6 for ZH→EN and AR→EN and 7 for DE→EN.[3] When instead tuning to maximize oracle BLEU for 20-diverse lists, the optimal n stayed at 7 for DE→EN, but increased to 7 for AR→EN and 9 for ZH→EN. These values are noticeably larger than n-gram sizes typically used in language modeling and evaluation. They suggest that for optimal oracle BLEU, translations with long-spanning amounts of repeated material should be avoided, while short overlapping n-grams are permitted.

Figure 1 shows other statistics on TEST for ZH→EN. Plots for AR→EN and DE→EN are qualitatively similar. We divided the TEST sentences into quartiles based on BLEU+1 of the 1-best translations from the baseline system. We computed the median, min, and max BLEU+1 on each list and averaged over the sentences in each quartile. As shown in the plot, the ranges of 20-diverse lists subsume those of 20-best lists, though the medians of diverse lists drop when the baseline system has high BLEU score. This matches intuition: when the baseline system is performing well, forcing it to find different translations is likely to result in worse translations. So we may expect diverse lists to be most helpful for more difficult sentences, a point we return to in our experiments below.

7 System Combination Experiments

One way to evaluate the quality of our diverse lists is to use them in system combination, as was similarly done by Devlin and Matsoukas (2012) and Cer et al. (2013). We use the system combination framework of Heafield and Lavie (2010b), which has an open-source implementation (Heafield and Lavie, 2010a).[4]

We use our baseline systems (trained on TUNE1) to generate lists for system combination on TUNE2 and TEST. We compare M-best lists, unique M-best lists, and M-diverse lists, with $M \in \{10, 15, 20\}$.[5] For each choice of list type and M, we trained the system combiner on TUNE2 and tested on TEST with the learned parameters. System combination hyperparameters (whether to use feature length normalization; the size of the k-best lists generated by the system combiner during tuning, $k \in \{300, 600\}$) were chosen to maximize BLEU on TUNE_{200}. Also, we removed the `individual` features from the default feature set because they correspond to individual systems in the combination; they did not seem appropriate for us since our hypotheses all come from the same system.

The results are shown in Table 2. Like Devlin and Matsoukas (2012), we see no gain from system combination using M-best lists. We see some improvement with unique lists, particularly for AR→EN, although it is not consistent across M values. But we see larger improvements with diverse lists for AR→EN and ZH→EN. For these language pairs, our

[3] The optimal values of λ were 0.005 for AR→EN and 0.01 for ZH→EN and DE→EN. Since these values depend on the scale of the weights learned by MERT, they are difficult to interpret in isolation.

[4] The implementation uses MERT to tune parameters, but we found this to be time-consuming and noisy for the larger feature sets. So we used a structured support vector machine learning framework instead (described in Section 8), using multiple iterations of learning interleaved with (system combiner) N-best list generation, and accumulating N-best lists across iterations.

[5] Dissimilarity hyperparameters n and λ were again chosen to maximize oracle BLEU on TUNE_{200}, separately for each M and for each language pair.

	AR→EN			ZH→EN			DE→EN		
	10	15	20	10	15	20	10	15	20
baseline (no system combination)	50.1			36.9			21.8		
M-best	50.2	50.1	50.0	36.7	36.9	37.0	21.7	21.7	21.8
unique M-best (from 1000-best list)	50.6	50.0	50.8	37.1	36.9	37.1	21.8	**21.9**	**21.9**
M-diverse	**51.4**	**51.2**	**51.2**	**37.6**	**37.6**	**37.5**	**22.0**	21.8	21.6

Table 2: System combination results (%BLEU on TEST). Size of lists is $M \in \{10, 15, 20\}$. Highest score in each column is bold.

	AR→EN				ZH→EN				DE→EN			
	q1	q2	q3	q4	q1	q2	q3	q4	q1	q2	q3	q4
baseline	30.1	44.1	55.1	**70.0**	15.2	28.9	41.0	**57.5**	5.3	14.4	23.7	**40.9**
15-best	30.1	44.6	55.5	68.8	15.9	29.2	40.5	56.8	6.0	15.0	23.6	40.0
unique 15-best	30.4	44.7	55.2	68.4	16.7	29.0	41.2	56.6	5.9	14.9	**23.8**	40.6
15-diverse	**31.3**	**45.3**	**57.8**	69.1	**17.7**	**30.6**	**41.7**	56.9	**7.6**	**15.2**	23.4	39.6

Table 3: System combination results (%BLEU on quartiles of TEST, $M = 15$). Source sentences were divided into quartiles (numbered "qn") according to BLEU+1 of the 1-best translations of the baseline system. Highest score in each column is bold.

gains are similar to those seen by Devlin and Matsoukas, but use our simpler dissimilarity function.[6] For DE→EN, results are similar for all settings and do not show much improvement from system combination.

In Table 3, we break down the scores according to 1-best BLEU+1 quartiles, as done in Figure 1.[7] In general, we find the largest gains for the low-BLEU translations. For the two worst BLEU quartiles, we see gains of 1.2 to 2.5 BLEU points, while the gains shrink or disappear entirely for the best quartile. This may be a worthwhile trade-off: a large improvement in the worst translations may be more significant to users than a smaller degredation on sentences that are already being translated well. In addition, quality estimation (Specia et al., 2011; Bach et al., 2011) could be used to automatically determine the BLEU quartile for each sentence. Then system combination of diverse translations might be used only when the 1-best translation is predicted to be of low quality.

8 Reranking Experiments

We now turn to discriminative reranking, which has frequently been used to easily add rich features to a model. It has been used for MT with varying degree of success (Och et al., 2004; Shen et al., 2004; Hildebrand and Vogel, 2008); some have attributed its mixed results to a lack of diversity in the M-best lists traditionally used. We propose diverse lists as a way to address this concern.

8.1 Learning Framework

Several learning formulations have been proposed for M-best reranking. One commonly-used approach in MT is MERT, used in the reranking experiments of Och et al. (2004) and Hildebrand and Vogel (2008), among others. We experimented with MERT and other algorithms, including pairwise ranking optimization (Hopkins and May, 2011), but we found best results using the approach of Yadollahpour et al. (2013), who used a slack-rescaled structured support vector machine (Tsochantaridis et al., 2005) with L2 regularization. As a sentence-level loss, we used negated BLEU+1. We used the 1-slack cutting-plane algorithm of Joachims et al. (2009) for optimization during learning.[8] A more detailed description of the reranker is provided in the supplementary material.

We used 5-fold cross-validation on TUNE2 to choose the regularization parameter C from the set $\{0.01, 0.1, 1, 10\}$. We selected the value yielding the highest average BLEU score across the held-out

[6] They reported +0.8 BLEU from system combination for AR→EN, and saw a further +0.5–0.7 from their new features.

[7] Quartile points are: 39, 49, 61 for AR→EN; 25, 36, and 47 for ZH→EN; and 14.5, 21.1, and 30.3 for DE→EN.

[8] Our implementation uses OOQP (Gertz and Wright, 2003) to solve the quadratic program in the inner loop, which uses HSL, a collection of Fortran codes for large-scale scientific computation (`www.hsl.rl.ac.uk`).

folds. This value was then used for one final round of training on the entirety of TUNE2. Additionally, we tuned the decision to return the parameters at convergence or those that produced the highest training corpus BLEU score. Since we use a sentence-level metric during training (BLEU+1) and a corpus-level metric for final evaluation (BLEU), we found that it was often better to return parameters that produced the highest training BLEU score.

This tuning procedure was repeated for each feature set and for each list type (M-best or diverse). The test set was not used for any of this tuning.

8.2 Features

In addition to the features from the baseline models (14 for phrase-based, 8 for hierarchical), we add 36 more for reranking:

Inverse Model 1 (INVMOD1): We added the "inverse" versions of the three IBM Model 1 features described in Section 2.2 of Hildebrand and Vogel (2008). The first is the probability of the source sentence given the translation under IBM Model 1, the second replaces the $\sum$ with a $\max$ in the first feature, and the third computes the percentage of words whose lexical translation probability falls below a threshold. We also include versions of the first 2 features normalized by the translation length, for a total of 5 INVMOD1 features.

Large LM (LLM): We created a large 4-gram LM by interpolating LMs from the WMT news data, Gigaword, Europarl, and the DE→EN news commentary (NC) corpus to maximize likelihood of a held-out development set (WMT08 test set). We used the average per-word log-probability as the single feature function in this category.

Syntactic LM (SYN): We used the syntactic treelet language model of Pauls and Klein (2012) to compute two features: the translation log probability and the length-normalized log probability.

Finite/Non-Finite Verbs (VERB): We ran the Stanford part-of-speech (POS) tagger (Toutanova et al., 2003) on each translation and added four features: the fraction of *words* tagged as finite/non-finite verbs, and the fraction of *verbs* that are finite/non-finite.[9]

[9]Words tagged as MD, VBP, VBZ, and VBD were counted as finite verbs, and VB, VBG, and VBN were non-finite verbs.

Reranking features	AR→EN best	AR→EN div	ZH→EN best	ZH→EN div	DE→EN best	DE→EN div
N/A (baseline)	50.1		36.9		21.8	
None	50.5	50.7	37.3	37.1	21.9	21.6
+ INVMOD1	50.3	50.8	**37.6**	37.1	22.0	21.8
+ LLM, SYN	50.5	51.1	37.4	37.3	21.7	21.7
+ VERB, DISC	50.4	51.3	37.3	37.3	21.9	22.2
+ GOOG	50.7	51.3	36.8	37.1	21.9	22.2
+ WCLM	51.2	**51.8**	37.3	37.4	22.2	**22.3**

Table 4: Reranking results (%BLEU on TEST).

Discriminative Word/Tag LMs (DISC): For each language pair, we generated 10,000-best lists for TUNE1 and computed BLEU+1 for each. From these lists, we estimated 3- and 5-gram LMs, weighting the n-gram counts by the BLEU+1 scores.[10] We repeated this procedure except using 1 *minus* BLEU+1 as the weight (learning a language model of "bad" translations). This yielded 4 features. The procedure was then repeated using POS tags instead of words, for 8 features in total.

Google 5-Grams (GOOG): Translations were compared to the Google 5-gram corpus (LDC2006T13) to compute: the number of 5-grams that matched, the number of 5-grams that missed, and a set of indicator features that fire if the fraction of 5-grams that matched in the sentence was greater than $\{0.05, 0.1, 0.2, \ldots, 0.9\}$, for a total of 12 features.

Word Cluster LMs (WCLM): Using an implementation provided by Liang (2005), we performed Brown clustering (Brown et al., 1992) on 900k English sentences, including the NC corpus and random sentences from Gigaword. We clustered words that appeared at least twice, once with 300 clusters and again with 1000. We then replaced words with their clusters in a large corpus consisting of the WMT news data, Gigaword, and the NC data. An additional cluster label was used for unknown words. For each of the clusterings (300 and 1000), we estimated 5- and 7-gram LMs with Witten-Bell smoothing (Witten and Bell, 1991). We added 4 features to the reranker, one for the log-probability of the translation under each of the word cluster LMs.

[10]Before estimating LMs, we projected the sentence weights so that the min and max per source sentence were 0 and 1.

List type	Features	
	None	All
20 best	50.3	50.6
100 best	50.6	50.8
200 best	50.4	51.2
1000 best	50.5	51.2
unique 20 best	50.5	51.2
unique 100 best	50.6	51.2
unique 200 best	50.4	51.3
20 diverse	50.5	51.1
20 div $\times$ 5 best	50.6	51.4
20 div $\times$ 10 best	**50.7**	51.3
20 div $\times$ 50 best	**50.7**	**51.8**

Table 5: List comparison for AR→EN reranking.

8.3 Results

Our results are shown in Table 4. We report results using the baseline system alone (labeled "N/A (baseline)"), and reranking standard M-best lists and our diverse lists. For diverse lists, we use the "20 div $\times$ 50 best" lists described in Section 5.3, with the tuned dissimilarity hyperparameters reported in Section 6. In the reranking settings, we also report results without adding any additional features (the row labeled "None").[11]

The remaining rows add features. For AR→EN, we see the largest gains, both over the baseline as well as differences between M-best lists and diverse lists. When using all features, we achieve a gain of 0.6 BLEU over M-best reranking and 1.7 BLEU points over the baseline system. The difference of 0.6 BLEU is consistent across feature subsets. We found the WCLM features to give the largest individual improvement, with the remaining feature sets each contributing a small amount. For Chinese and German, the gains and individual differences are smaller. Nonetheless, diverse lists appear to be more robust for these language pairs as features are added.

In Table 5, we compare several sizes and types of lists for AR→EN reranking both with no additional features and with the full set. We see that using 20-diverse lists nearly matches the performance of 200-best lists. Also, retaining 50-best lists for each diverse solution improves BLEU by 0.7.

[11]Though such results have not always been reported in prior work on reranking, we generally found them to improve over the baseline, presumably because seeing more data improves generalization ability.

		Train	
		best	div
Test	best	51.2	51.7
	div	50.5	51.8

Table 6: Comparing M-best and diverse lists for training/testing (AR→EN, all features).

Thus far, when training the reranker on M-best lists, we tested it on M-best lists, and similarly for diverse lists. Table 6 shows what happens with the other two pairings for AR→EN with the full feature set. When training on diverse lists, we see very little difference in BLEU whether testing on M-best or diverse lists. This has a practical benefit: we can use (computationally-expensive) diverse lists during offline training and then use fast M-best lists at test time. When training on M-best lists and *testing* on diverse lists, we see a substantial drop (51.2 vs 50.5). The reranker may be overfitting to the limited scope of translations present in typical M-best lists, thereby hindering its ability to correctly rank diverse lists at test time. These results suggest that part of the benefit of using diverse lists comes from seeing a larger portion of the output space during training.

9 Human Post-Editing Experiments

We wanted to determine whether diverse translations could be helpful to users struggling to understand the output of an imperfect MT system. We consider a post-editing task in which users are presented with translation output without the source sentence, and are asked to improve it. This setting has been studied; e.g., Koehn (2010) presented evidence that monolingual speakers could often produce improved translations for this task, occasionally reaching the level of an expert translator.

Here, we use a novel variation of this task in which *multiple* translations are shown to editors. We compare the use of entries from an M-best list and entries from a diverse list. Again, the original source sentence is not provided. Our goal is to determine whether multiple, diverse translations can help users to more accurately guess the meaning of the original sentence than entries from a standard M-best list. If so, commercial MT systems might permit users to request additional diverse translations for those sentences whose model-best translations are difficult to understand.

9.1 Translation List Post-Editing

We use Amazon Mechanical Turk (MTurk) for this experiment. Workers are shown 3 outputs from an MT system. They are not shown the original sentence, nor are they shown a reference. Based on the 3 imperfect translations, they are asked to write a single fluent English translation that best captures the understood meaning. Half of the time, the worker is shown 3 entries from an M-best list, and the other half of the time 3 entries from a diverse list. We then compare the outputs produced under the two conditions. The goal is to measure whether workers are able to produce translations that are closer in meaning to the (unseen) references when shown diverse translations. We refer to this task as the EDITING task.

To evaluate the outputs, we use a second task in which users are shown a reference translation along with two outputs from the first task: one created from M-best lists and one from diverse lists. Workers in this task are asked to choose which translation is a better match to the reference in terms of meaning, or they can indicate that the translations are of the same quality. We refer to this second task as the EVAL task.

9.2 Dissimilarity Functions

To generate diverse lists for the EDITING task, we use the same dissimilarity function as in reranking, but we tune the hyperparameters n and λ differently. Since our expectation here is that workers may combine information from multiple translations to produce a superior output, we are interested in the *coverage* of the translations in the diverse list, rather than the oracle BLEU score.

We designed a metric based on coverage of entire lists of translations. It is similar to BLEU+1, except (1) it uses n-gram recalls instead of n-gram precisions, (2) there is no brevity penalty term, and (3) it compares a *list* to a set of references and any translation in the list can contribute a match of an n-gram in any reference. Like BLEU, counts are clipped based on those in the references. We maximized this metric over diverse lists of length 5, for $n \in \{2, 3, \ldots, 9\}$ and $\lambda \in \{0.005, 0.01, 0.05, 0.1, 0.2\}$. The optimal values for AR→EN were $n = 4$ and $\lambda = 0.1$, while for ZH→EN they were $n = 4$ and $\lambda = 0.2$. These n values are smaller than for reranking, and the λ values are larger. This suggests that, when maximizing coverage of a small diverse list, more dissimilarity is desired among the translations.

9.3 Detailed Procedure

We focused on AR→EN and ZH→EN for this study. We sampled 200 sentences from their test sets, chosen from among those whose reference translation was between 5 and 25 words. We generated a unique 5-best list for each sentence using our baseline system (described in Section 5.2) and also generated a diverse list of length 5 using the dissimilarity function Δ with hyperparameters tuned using the procedure from the previous section. We untokenized and truecased the translations. We dropped non-ASCII characters because we feared they would confuse our workers. As a result, workers must contend with missing words in the output, often proper nouns.

Given the 2 lists for each sentence, we sampled two integers $i, j \in \{2, 3, 4, 5\}$ without replacement. The indices i and j indicate two entries from the lists. We took translations 1, i, and j from the 5-best list and created an EDITING task from them. We did the same using entries 1, i, and j from the diverse list. We repeated this process 3 times for each sentence, obtaining $3 \times 2 = 6$ tasks for each, giving us a total of 1,200 EDITING tasks per language pair.

The outputs of the EDITING tasks were evaluated with EVAL tasks. For each sentence, we had 3 post-edited outputs generated using entries in 5-best lists and 3 post-edited outputs from diverse lists. We created EVAL tasks for all 9 output pairs, for all 200 sentences per language pair. We additionally gave each task to three MTurk workers. This gave us 10,800 evaluation judgments for the EVAL task.

9.4 Results

Figure 2 shows the quartile breakdown for judgments collected from the EVAL task. The Y axis represents the percentage of judgments for which best/diverse outputs were preferred; the missing percentage for each bin is accounted for by "same" judgments.

We observe an interesting phenomenon. Overall, there is a slight preference for the post-edited outputs of M-best entries ("best") over those from diverse translations ("div"); this preference is clearest

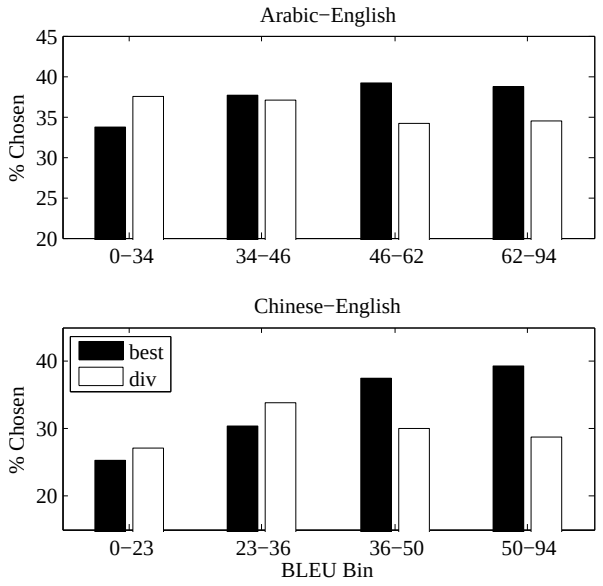

Figure 2: Percentages in which post-edited output given M-best entries ("best") was preferred by human evaluators as compared to post-edited output given diverse translations ("div"), broken down by the BLEU+1 score of the 1-best translation for the sentences. When the baseline system is doing poorly, diversity helps post-editors to produce better translations.

when the baseline system's 1-best translation had a high BLEU score. However, we see this trend reversed for sentences in which the baseline system's 1-best translation had a low BLEU score. In general, when the BLEU score of the baseline system is below 35, it is preferable to give diverse translations to users for post-editing. But when the baseline system does very well, diverse translations do not contribute anything, and in fact hurt because they may distract users from the high-quality (and typically very similar) translations from the 5-best lists.

Estimation of the quality of the output ("confidence estimation") has recently gained interest in the MT community (Specia et al., 2011; Bach et al., 2011; Callison-Burch et al., 2012; Bojar et al., 2013), including specifically for post-editing (Tatsumi, 2009; Specia, 2011; Koponen, 2012). Future work could investigate whether such automatic confidence estimation could be used to identify situations in which diverse translations can be helpful for aiding user understanding.

10 Future Work

Our dissimilarity function captures diversity in the particular phrases used by an MT system, but for certain applications we may prefer other types of diversity. Defining the dissimilarity function on POS tags or word clusters would help us to capture stylistic patterns in sentence structure, as would targeting syntactic structures in syntax-based translation.

A weakness of our approach is its computational expense; by contrast, the method of Devlin and Matsoukas (2012) obtains diverse translations more efficiently by extracting them from a single decoding of an input sentence (albeit with a wide beam). We expect their ideas to be directly applicable to our setting in order to get diverse solutions more cheaply. We also plan to explore methods of explicitly targeting multiple, diverse solutions as part of the search algorithm.

Finally, M-best lists are currently used to approximate structured spaces for many areas of MT, including tuning (Och, 2003), minimum Bayes risk decoding (Kumar and Byrne, 2004), and pipelines (Venugopal et al., 2008). Future work could replace M-best lists with diverse lists in these and related tasks, whether for MT or other areas of structured NLP.

Acknowledgments

We thank the anonymous reviewers as well as Colin Cherry, Kenneth Heafield, Silja Hildebrand, Fei Huang, Dan Klein, Adam Pauls, and Bing Xiang. DB was partially supported by the National Science Foundation under Grant No. 1353694.

References

N. Bach, F. Huang, and Y. Al-Onaizan. 2011. Goodness: A method for measuring machine translation confidence. In *Proc. of ACL.*

D. Batra, P. Yadollahpour, A. Guzman-Rivera, and G. Shakhnarovich. 2012. Diverse M-best solutions in Markov random fields. In *Proc. of ECCV.*

O. Bojar, C. Buck, C. Callison-Burch, C. Federmann, B. Haddow, P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proc. of WMT.*

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based N-gram mod-

els of natural language. *Computational Linguistics*, 18.

C. Callison-Burch, P. Koehn, C. Monz, and O. Zaidan. 2011. Findings of the 2011 Workshop on Statistical Machine Translation. In *Proc. of WMT*.

C. Callison-Burch, P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia. 2012. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proc. of WMT*.

D. Cer, C. D. Manning, and D. Jurafsky. 2013. Positive diversity tuning for machine translation system combination. In *Proc. of WMT*.

P. Chang, M. Galley, and C. D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proc. of WMT*.

E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proc. of ACL*.

S. Chatterjee and N. Cancedda. 2010. Minimum error rate training by sampling the translation lattice. In *Proc. of EMNLP*.

S. Chen and J. Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical report 10-98, Harvard University.

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).

M. Collins and T. Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1).

M. Collins. 2000. Discriminative reranking for natural language parsing. In *Proc. of ICML*.

J. DeNero, D. Chiang, and K. Knight. 2009. Fast consensus decoding over translation forests. In *Proc. of ACL*.

J. Devlin and S. Matsoukas. 2012. Trait-based hypothesis selection for machine translation. In *Proc. of NAACL*.

C. Dyer. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *Proc. of HLT-NAACL*.

C. Dyer. 2010. *A Formal Model of Ambiguity and its Applications in Machine Translation*. Ph.D. thesis, University of Maryland.

J. R. Finkel, C. D. Manning, and A. Y. Ng. 2006. Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proc. of EMNLP*.

E. M. Gertz and S. J. Wright. 2003. Object-oriented software for quadratic programming. *ACM Transactions on Mathematical Software*, 29(1).

J. Gillenwater, A. Kulesza, and B. Taskar. 2012. Discovering diverse and salient threads in document collections. In *Proc. of EMNLP*.

K. Heafield and A. Lavie. 2010a. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *The Prague Bulletin of Mathematical Linguistics*, 93.

K. Heafield and A. Lavie. 2010b. Voting on n-grams for machine translation system combination. In *Proc. of AMTA*.

K. Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proc. of WMT*.

A. Hildebrand and S. Vogel. 2008. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *Proc. of AMTA*.

H. Hoang, P. Koehn, and A. Lopez. 2009. A Unified Framework for Phrase-Based, Hierarchical, and Syntax-Based Statistical Machine Translation. In *Proc. of IWSLT*.

M. Hopkins and J. May. 2011. Tuning as ranking. In *Proc. of EMNLP*.

L. Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proc. of ACL*.

T. Joachims, T. Finley, and C. Yu. 2009. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1).

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL (demo session)*.

P. Koehn. 2010. Enabling monolingual translators: Post-editing vs. options. In *Proc. of NAACL*.

M. Koponen. 2012. Comparing human perceptions of post-editing effort with post-editing operations. In *Proc. of WMT*.

A. Kulesza and B. Taskar. 2010. Structured determinantal point processes. In *Proc. of NIPS*.

A. Kulesza and B. Taskar. 2011. Learning determinantal point processes. In *Proc. of UAI*.

S. Kumar and W. Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proc. of HLT-NAACL*.

S. Kumar, W. Macherey, C. Dyer, and F. Och. 2009. Efficient minimum error rate training and minimum

Bayes-risk decoding for translation hypergraphs and lattices. In *Proc. of ACL-IJCNLP*.

Y. Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan. 2003. Language model based Arabic word segmentation. In *Proc. of ACL*.

Z. Li, J. Eisner, and S. Khudanpur. 2009. Variational decoding for statistical machine translation. In *Proc. of ACL*.

P. Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

C. Lin and F. J. Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proc. of COLING*.

W. Macherey and F. J. Och. 2007. An empirical study on computing consensus translations from multiple machine translation systems. In *Proc. of EMNLP-CoNLL*.

W. Macherey, F. J. Och, I. Thayer, and J. Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proc. of EMNLP*.

F. J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of ACL*.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).

F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. A smorgasbord of features for statistical machine translation. In *HLT-NAACL*.

F. J. Och. 2003. Minimum error rate training for statistical machine translation. In *Proc. of ACL*.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*.

A. Pauls and D. Klein. 2012. Large-scale syntactic language modeling with treelets. In *Proc. of ACL*.

A.-V. Rosti, N. F. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. 2007. Combining outputs from multiple machine translation systems. In *HLT-NAACL*.

L. Shen and A. K. Joshi. 2003. An SVM-based voting algorithm with application to parse reranking. In *Proc. of CoNLL*.

L. Shen, A. Sarkar, and F. J. Och. 2004. Discriminative reranking for machine translation. In *Proc. of HLT-NAACL*.

L. Specia, N. Hajlaoui, C. Hallett, and W. Aziz. 2011. Predicting machine translation adequacy. In *Proc. of MT Summit XIII*.

L. Specia. 2011. Exploiting objective annotations for measuring translation post-editing effort. In *Proc. of EAMT*.

A. Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *Proc. of ICSLP*.

M. Tatsumi. 2009. Correlation between automatic evaluation metric scores, post-editing speed, and some other factors. In *Proc. of MT Summit XII*.

K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of HLT-NAACL*.

R. Tromble, S. Kumar, F. J. Och, and W. Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proc. of EMNLP*.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. Large margin methods for structured and interdependent output variables. *JMLR*, 6.

A. Venugopal, A. Zollmann, N.A. Smith, and S. Vogel. 2008. Wider pipelines: N-best alignments and parses in MT training. In *Proc. of AMTA*.

I. H. Witten and T. C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4).

T. Xiao, J. Zhu, and T. Liu. 2013. Bagging and boosting statistical machine translation systems. *Artif. Intell.*, 195.

P. Yadollahpour, D. Batra, and G. Shakhnarovich. 2013. Discriminative re-ranking of diverse segmentations. In *Proc. of CVPR*.

A Systematic Exploration of Diversity in Machine Translation
Supplementary Material

Kevin Gimpel[*] **Dhruv Batra**[†] **Chris Dyer**[‡] **Gregory Shakhnarovich**[*]
[*]Toyota Technological Institute at Chicago, Chicago, IL 60637, USA
[†]Virginia Tech, Blacksburg, VA 24061, USA
[‡]Carnegie Mellon University, Pittsburgh, PA 15213, USA
Corresponding author: kgimpel@ttic.edu

1 Structured Support Vector Machine Reranking

We now describe the reranking algorithm of Yadollahpour et al. (2013) and discuss how we applied it to MT.

Let $\mathbf{Y}_i = \{\mathbf{y}_i^{(1)} \ldots \mathbf{y}_i^{(M)}\}$ denote the set of M translations for source sentence x_i, and $\mathbf{Y}_i^R$ the set of reference translations.[1] Note that frequently it is the case that $\mathbf{Y}_i \cap \mathbf{Y}_i^R = \varnothing$. Let $\mathbf{y}_i^*$ denote the highest-quality translation in the set, i.e., $\mathbf{y}_i^* = \text{argmin}_{\mathbf{y} \in \mathbf{Y}_i} \ell(\mathbf{Y}_i^R, \mathbf{y})$, where $\ell(\mathbf{Y}_i^R, \mathbf{y})$ is the negated BLEU+1 score (Lin and Och, 2004) of $\mathbf{y}$. The quality of solution $\mathbf{y}_i^*$ leads to an upper-bound on the reranker performance since we must select one solution from $\mathbf{Y}_i$.

The reranking model assigns a score S_r to each translation in the set, i.e., $S_r(\mathbf{y}) = \boldsymbol{\alpha}^\intercal \boldsymbol{\psi}(x, \mathbf{y})$, where $\boldsymbol{\alpha}$ and $\boldsymbol{\psi}(x, \mathbf{y})$ are the reranker weights and features respectively. The reranking features can be quite complex, as decoding simply finds the highest scoring solution in the set: $\hat{\mathbf{y}}_i = \text{argmax}_{\mathbf{y} \in \mathbf{Y}_i} S_r(\mathbf{y})$.

The objective of the reranker (picking the best solution $\mathbf{y}_i^*$ from the set) is formulated as a structured SVM (Tsochantaridis et al., 2005). For training, we use the following loss function, defined for a hypothesis $\hat{\mathbf{y}}_i$:

$$\mathcal{L}(\mathbf{Y}_i^R, \hat{\mathbf{y}}_i) = \ell(\mathbf{Y}_i^R, \hat{\mathbf{y}}_i) - \ell(\mathbf{Y}_i^R, \mathbf{y}_i^*), \quad (1)$$

i.e., the negated BLEU+1 score of translation $\hat{\mathbf{y}}_i$ *relative* to that of the best translation in this set ($\mathbf{y}_i^*$). This relative loss forces the reranker to focus its effort on training instances where it is underperforming w.r.t. the set, rather than in absolute terms. For instance, consider two input sentences i, j with two translations each. The translations for sentence i have BLEU+1 scores of 65 and 45 while the translations for sentence j have scores 40 and 35. Using only absolute ℓ in the reranking objective would emphasize sentence j, since both translations for j have low BLEU compared to translations for i. Using the relative loss correctly shifts the focus to i because an incorrect choice in that set is much costlier (difference of 20 points) than an incorrect choice in set j (5 points).

We learn the reranker parameters $\boldsymbol{\alpha}$ by solving the following quadratic program:

$$\min_{\boldsymbol{\alpha}, \xi_i} \ \|\boldsymbol{\alpha}\|_2^2 + C \sum_{i \in [n]} \xi_i \quad (2a)$$

$$\text{s.t. } \boldsymbol{\alpha}^\intercal \Big(\boldsymbol{\psi}(x_i, \mathbf{y}_i^*) - \boldsymbol{\psi}(x_i, \mathbf{y}) \Big) \geq 1 - \frac{\xi_i}{\mathcal{L}(\mathbf{Y}_i^R, \mathbf{y})} \quad (2b)$$

$$\xi_i \geq 0, \qquad \forall \mathbf{y} \in \mathbf{Y}_i \setminus \mathbf{y}_i^*, \quad (2c)$$

Intuitively, (2b) maximizes the (soft) margin between the score of the oracle solution and all other solutions in the set. The violation in the margin ξ_i is scaled by the loss of the solution. Thus if in addition to $\mathbf{y}_i^*$ there are other good solutions in the set, the margin for such solutions will not be tightly enforced. On the other hand, the margin between $\mathbf{y}_i^*$ and bad solutions will be very strictly enforced. We solve (2) via the 1-slack cutting-plane algorithm of Joachims et al. (2009). We used OOQP (Gertz and Wright, 2003) to solve the quadratic program

[1] For conciseness, we do not explicitly show the derivation variable $\mathbf{h}_i^{(j)}$ associated with translation j, but it is always available for computing features for $\mathbf{y}_i^{(j)}$.

in the inner loop, which uses HSL, a collection of Fortran codes for large-scale scientific computation (`www.hsl.rl.ac.uk`).

References

E. M. Gertz and S. J. Wright. 2003. Object-oriented software for quadratic programming. *ACM Transactions on Mathematical Software*, 29(1).

T. Joachims, T. Finley, and C. Yu. 2009. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1).

C. Lin and F. J. Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proc. of COLING*.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. Large margin methods for structured and interdependent output variables. *JMLR*, 6.

P. Yadollahpour, D. Batra, and G. Shakhnarovich. 2013. Discriminative re-ranking of diverse segmentations. In *Proc. of CVPR*.