Trait-Based Hypothesis Selection For Machine Translation

Jacob Devlin and **Spyros Matsoukas**
Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA 02138, USA
{jdevlin,smatsouk}@bbn.com

Abstract

In the area of machine translation (MT) system combination, previous work on generating input hypotheses has focused on varying a core aspect of the MT system, such as the decoding algorithm or alignment algorithm. In this paper, we propose a new method for generating diverse hypotheses from a *single* MT system using *traits*. These traits are simple properties of the MT output such as "average output length" and "average rule length." Our method is designed to select hypotheses which vary in trait value but do not significantly degrade in BLEU score. These hypotheses can be combined using standard system combination techniques to produce a 1.2-1.5 BLEU gain on the Arabic-English NIST MT06/MT08 translation task.

1 Introduction

In Machine Translation (MT), the output from multiple decoding systems can be used to create a new output which is better than any single input system, using a procedure known as *system combination*.

Normally, the input systems are generated by varying some important aspect of the MT system, such as the alignment algorithm (Xu and Rosti, 2010) or tokenization algorithm (de Gispert et al., 2009). Unfortunately, creating novel algorithms to perform some important aspect of MT decoding is obviously quite challenging. Thus, it is difficult to increase the number of input systems in a meaningful way.

In this paper, we show it is possible to create diverse input hypotheses for combination *without* making any algorithmic changes. Instead, we use *traits*, which are very simple attributes of the MT output, such as "output length" and "average rule length." Our basic procedure is to intelligently select hypotheses from our decoding forest which vary in trait value, but have minimal BLEU degradation compared to our baseline. We then combine these to produce a substantial gain. Note that all of the hypotheses are generated from a *single* decode of a *single* input system.

Additionally, our method is completely compatible with multi-system combination, since our procedure can be applied to each input system, and then these systems can be combined as normal.

Methods for automatically creating diverse hypotheses from a single system have been explored in speech recognition (Siohan et al., 2005), but we know of no analogous work applied to machine translation. Our procedure does share some surface similarities with techniques such as variational decoding (VD) (Li et al., 2009), but the goal in those techniques is to find output which is consistent with the entire forest, rather than to select hypotheses with particular attributes. In fact, VD can be applied in conjunction by running VD on the rescored forest

This work was supported by DARPA/I2O Contract No. HR0011-12-C-0014 under the BOLT program (Approved for Public Release, Distribution Unlimited). The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

for each trait condition.[1]

2 Description of MT System

Our machine translation system is a string-to-dependency hierarchical decoder based on (Shen et al., 2008) and (Chiang, 2007). Bottom-up chart parsing is performed to produce a shared forest of derivations. The decoder uses a log-linear translation model, so the score of derivation d is defined as:

$$S_d(\vec{w}) = \sum_{i=1}^{m} w_i \sum_{r \in R(d)} F_{ri} \quad (1)$$

where $R(d)$ is the set of translation rules that make up derivation d, m is the number of features, F_{ri} is the score of the i^{th} feature in rule r, and w_i is the weight of feature i. This weight vector is optimized discriminatively to maximize BLEU score on a tuning set, using the Expected-BLEU optimization procedure (Rosti et al., 2010).

Our decoder uses all of the standard statistical MT features, such as the language model, rule probabilities, and lexical probabilities. Additionally, we use 50,000 sparse, binary-valued features such as "Is the bi-gram 'united states' present in the output?", based on (Chiang et al., 2009). We use a 3-gram LM for decoding and a 5-gram LM for rescoring.

3 Trait Features

An MT *trait* represents a high-level property of the MT output.

The traits used in this paper are:

- *Null Source Words* – The percentage of source content words which align to null, i.e., are not translated.
- *Source Reorder* – The percentage of source terminals/non-terminals which cross alignment links inside their decoding rule.
- *Ngram Frequency* – The percentage of target 3-grams which are seen more than 10 times in the monolingual training.
- *Rule Frequency* – The percentage of rules which are seen more than 3 times in the parallel training.
- *Rule Length* – The average number of target words per rule.
- *Output Length* – The ratio of the number of target words in the MT output divided by the number of source words in the input.
- *High Lex Prob* – The percentage of source words which have a lexical translation probability greater than 0.1.

Each trait can be represented as the ratio of two linear decoding features. For example, for the *Output Length* trait, the "numerator" feature is the number of target words in the hypothesis, while the "denominator" feature is the number of source words in the input sentence. We can sum these feature scores over a test set, and the resulting quotient is the *Output Length* for that set.

Intuitively, each trait is associated with a particular *tradeoff*, such as fluency/adequacy or precision/recall. For example, when MT performance is maximized, shorter output tends to have higher precision but lower recall than longer output. For the *Ngram Frequency* trait, a greater percentage of high-frequency n-grams tends to result in more fluent but less adequate output. Similar intuitive justifications should be evident for the remaining traits.

4 Hypothesis Generation

The main goal of this work is to generate additional hypotheses which vary in trait values, while minimizing degradation to the BLEU score. So, imagine that we have some baseline MT output. Then, we want to generate a second set of hypotheses which have maximal BLEU score, *subject to the constraint* that the output must be 5% shorter.[2]

The question then becomes how to figure out which 5% of words should be removed. Rather than attempting to do this with a new algorithm, we simply let our existing MT models do it for us, using our standard optimization procedure. This is the essential purpose of the trait features – using the *Output Length* feature, the optimizer has a "knob" with which it can control the trait value independently of everything else.[3] Thus, the new hypotheses that

[1] We do not use VD here because we have not found it to be beneficial to our system.

[2] Note that the trait value is always aggregated over the entire set, and not computed sentence-by-sentence.

[3] A feature representing the number of words already exists in our baseline system, but no such feature exists for the other 6

we select are "optimal" in terms of our existing MT model probabilities, but have trait values which vary from the baseline in a precise way.

4.1 Optimization Function

Our normal optimization procedure uses n-best-based Expected-BLEU tuning (Rosti et al., 2010), which is a differentiable approximation of Maximum-BLEU tuning. To "target" a particular trait value, we add a second term equal to the squared error between the current trait value and the target trait value. Our modified optimization function which we seek to maximize is then:

$$Obj(\vec{w}) = ExpBLEU(\vec{w}) - \alpha \left(\frac{N(\vec{w})}{D(\vec{w})} - \tau\gamma \right)^2$$

where $\vec{w}$ is the MT feature weight vector, α is the weight of the trait term, γ is the baseline value of the trait, and τ is the "target" trait multiplier, $N(\vec{w})$ is the expected-value of the numerator feature, and $D(\vec{w})$ is the expected value of the denominator feature.

To give an example, imagine that for our baseline tune set the *Output Length* ratio is 1.2, and we want to create a hypothesis set with 5% fewer words. In that case, we would set $\gamma = 1.2$ and $\tau = 0.95$, so the target trait value is 1.14. We fix the free parameter α to 10, which forces the optimized trait value to be very close to the target.[4]

The trait-value functions $N(\vec{w})$ and $D(\vec{w})$ are computed as standard expected value functions, e.g.:

$$N(\vec{w}) = \sum_i \sum_j p_{ij}(\vec{w}) N_{ij}$$

where $p_{ij}(\vec{w})$ is the posterior probability of the j^{th} hypothesis of sentence i, and N_{ij} is the value of the numerator feature for hypothesis ij.[5]

4.2 Meta-Optimization

It is somewhat problematic to use a fixed multiplier τ on all of the traits, since on some traits it may cause a larger degradation than others. So, we take the reverse approach – for some targeted BLEU loss β, we find the maximum (or minimum) value of τ which causes a loss no greater than β, as computed on a held-out portion of the tune set.[6] Here, we find the maximum and minimum trait value for $\beta = 0.5$ and $\beta = 2.0$, resulting in 4 sets of weights per trait.

We can find the optimal τ for each β by performing a binary search on τ, where we run our optimization procedure and then compute the BLEU loss at each iteration.

4.3 Forest-Based Optimization

Since we have 7 traits, and we generate 4 sets of weights per trait, we have 28 "systems" to combine. Obviously, running 28 full decodes on each new test sentence is highly undesirable.

We resolve this issue by using our baseline derivation forest for both optimization and hypothesis generation. We perform a single round of decoding to generate a forest, and then perform iterative n-best optimization by rescoring the forest rather than re-decoding from scratch. [7] We constrain the 50,000 sparse feature weights to be fixed at their baseline values, to prevent over-fitting.

Once the weight sets are generated, the hypotheses for each trait condition can be generated by rescoring the forest inside of the decoder. Therefore, all 28 trait hypotheses can be generated for almost no cost over a single decode.

It should be noted we have found it beneficial to relax our MT pruning parameters in order to create a larger forest. This results in decoding which is roughly 2x-3x as slow as the baseline, and requires storing the larger forest in memory. However, we have found that the procedure still works well even with the standard pruning parameters. Additionally, we are investigating methods for diversifying the forest with less of a slowdown to decoding.

5 Combination

Once the different trait hypotheses have been generated, system combination can be performed using any method.

Here, we use a confusion network decoder based on (Rosti et al., 2010). The basic procedure is to

traits.

[4]Note that the $ExpBLEU(\vec{w})$ is raw BLEU *not* BLEU percentage, i.e., it's 0.4528 not 45.28

[5]$p_{ij}(\vec{w})$ is computed the same way as in $ExpBLEU(\vec{w})$. See (Rosti et al., 2010) for details.

[6]For example if the held-out baseline BLEU is 40.0 and $\beta = 0.5$, the BLEU after trait optimization can be no less than 39.5.

[7]Forest-based optimization such as (Pauls et al., 2009) could be used instead.

select one hypothesis as the "skeleton" and then incrementally align the remaining hypotheses to create a confusion network. The confusion network is decoded using an arc-level confidence score for each input system and a language model, the weights for which are estimated discriminatively to maximize BLEU.

6 Results

We present MT results in Table 1. Our experimental setup is compatible with the NIST MT08 constrained track. We trained our translation model on 35 million words of parallel data and our language model on 3.8 billion words of monolingual data. We use a portion of MT02-05 for tuning the MT baseline and the trait systems, and another portion of MT02-05 for tuning system combination.

We present results on Arabic-English MT06-newswire and MT08-eval. The systems were tuned and evaluated using IBM-BLEU. Our baseline system is 1.5 BLEU better than the best result from the NIST M08 evaluation.

For the *Trait Feats* condition, we simply added the numerator and denominator features for all 7 traits to the baseline system and re-optimized.[8] Somewhat surprisingly, this produces an 0.5-0.7 BLEU gain on its own. In this condition, although we do not target any particular trait values, the optimizer will naturally fine-tune the trait values to whatever is optimal for BLEU score. For example, the MT08 baseline value of *Source Reorder* is 0.307, while for the *Trait Feats* it is 0.330, so the system determined it is "optimal" to have 7.5% (0.330/0.307) more re-ordering than the baseline.

For the *Trait Comb* condition, we generated 28 trait hypothesis sets using the decoding forest from the *Trait Feats* condition. We combined these with the *Trait Feats* output using consensus network decoding. This produces an additional 0.8 BLEU gain, resulting in a 1.2-1.5 BLEU gain over the baseline.

We also present another condition, n*-best Comb*, where we perform confusion network combination on the 28-best hypotheses from *Trait Feats*. This represents the simplest and most trivial method of hypothesis selection. We observe no gain in BLEU on this condition. Other simple methods of hypotheses selection, such as optimizing systems to be "different" from one another (i.e., have high inter-system TER), also produced no gain over the single system. We include these results simply to demonstrate that it is *not* trivial to select hypotheses from a single system which produce a significant improvement in from system combination.

[8]Including the 50k sparse features.

	MT06 nw		**MT08 eval**	
	BLEU	Len	BLEU	Len
Baseline	55.11	99.1%	46.75	96.1%
Trait Feats	55.79*	99.3%	47.23*	96.0%
+n-best Comb	55.65	99.3%	47.24	96.2%
+Trait Comb	**56.65****	99.3%	**48.00****	96.2%

Table 1: Results on Arabic-English MT. * = Significant improvement at 95% confidence, as defined by (Koehn, 2004). ** = Significant improvement at 99.9% confidence. **BLEU** = IBM-BLEU score. **Len** = Hypothesis-to-reference length ratio.

7 Conclusions and Future Work

We demonstrated a method of intelligently selecting hypotheses from a decoding forest which can be combined with the baseline hypotheses to produce a significant gain in BLEU score. In the future, we plan to explore more trait types and alternate methods of system combination.

One possible application of this work is in fielded translation systems. Because our method produces high-quality complementary hypotheses at a low computational cost, the system could present these to the user as alternate translations. Going further, a user could prefer a particular output type, such as the fluency-tuned condition, and set that to be their default translation.

The major open question is how our trait-based combination interacts with multi-system combination. Imagine there are three different types of decoders which can be combined to produce some gain in the baseline condition. If you independently improve all three using trait-based combination, will the relative gain from multi-system combination be reduced? Or can you jointly combine *all* of the trait hypotheses and get an even greater relative gain? We plan to thoroughly explore this in the future.

References

D. Chiang, K. Knight, and W. Wang. 2009. 11,001 new features for statistical machine translation. In *NAACL*, pages 218–226.

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

A. de Gispert, S. Virpioja, M. Kurimo, and W. Byrne. 2009. Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *NAACL*, pages 73–76.

P. Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Z. Li, J. Eisner, and S. Khudanpur. 2009. Variational decoding for statistical machine translation. In *ACL*, pages 593–601.

A. Pauls, J. DeNero, and D. Klein. 2009. Consensus training for consensus decoding in machine translation. In *EMNLP*, pages 1418–1427.

A. Rosti, B. Zhang, S. Matsoukas, and R. Schwartz. 2010. BBN system description for WMT10 system combination task. In *WMT/MetricsMATR*, pages 321–326.

L. Shen, J. Xu, and R. Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL-HLT*, pages 577–585.

O. Siohan, B. Ramabhadran, and B. Kingsbury. 2005. Constructing ensembles of ASR systems using randomized decision trees. In *ICASSP*.

J. Xu and A. Rosti. 2010. Combining unsupervised and supervised alignments for MT: An empirical study. In *EMNLP*, pages 667–673.