Model Combination for Machine Translation

John DeNero,
UC Berkeley
denero@berkeley.edu

Shankar Kumar, Ciprian Chelba, and **Franz Och**
Google, Inc.
{shankarkumar,ciprianchelba,och}@google.com

Abstract

Machine translation benefits from two types of decoding techniques: consensus decoding over multiple hypotheses under a single model and system combination over hypotheses from different models. We present *model combination*, a method that integrates consensus decoding and system combination into a unified, forest-based technique. Our approach makes few assumptions about the underlying component models, enabling us to combine systems with heterogenous structure. Unlike most system combination techniques, we reuse the search space of component models, which entirely avoids the need to align translation hypotheses. Despite its relative simplicity, model combination improves translation quality over a pipelined approach of first applying consensus decoding to individual systems, and then applying system combination to their output. We demonstrate BLEU improvements across data sets and language pairs in large-scale experiments.

1 Introduction

Once statistical translation models are trained, a decoding approach determines what translations are finally selected. Two parallel lines of research have shown consistent improvements over the standard max-derivation decoding objective, which selects the highest probability derivation. *Consensus decoding* procedures select translations for a single system by optimizing for model predictions about n-grams, motivated either as minimizing Bayes risk (Kumar and Byrne, 2004), maximizing sentence similarity (DeNero et al., 2009), or approximating a max-translation objective (Li et al., 2009b). *System combination* procedures, on the other hand, generate translations from the output of multiple component systems (Frederking and Nirenburg, 1994). In this paper, we present *model combination*, a technique that unifies these two approaches by learning a consensus model over the n-gram features of multiple underlying component models.

Model combination operates over the component models' posterior distributions over translation derivations, encoded as a forest of derivations.[1] We combine these components by constructing a linear consensus model that includes features from each component. We then optimize this consensus model over the space of all translation derivations in the support of all component models' posterior distributions. By reusing the components' search spaces, we entirely avoid the hypothesis alignment problem that is central to standard system combination approaches (Rosti et al., 2007).

Forest-based consensus decoding techniques differ in whether they capture model predictions through n-gram posteriors (Tromble et al., 2008; Kumar et al., 2009) or expected n-gram counts (DeNero et al., 2009; Li et al., 2009b). We evaluate both in controlled experiments, demonstrating their empirical similarity. We also describe algorithms for expanding translation forests to ensure that n-grams are local to a forest's hyperedges, and for exactly computing n-gram posteriors efficiently.

Model combination assumes only that each translation model can produce expectations of n-gram features; the latent derivation structures of component systems can differ arbitrarily. This flexibility allows us to combine phrase-based, hierarchical, and syntax-augmented translation models. We evaluate by combining three large-scale systems on Chinese-English and Arabic-English NIST data sets, demonstrating improvements of up to 1.4 BLEU over the

[1] In this paper, we use the terms *translation forest* and *hypergraph* interchangeably.

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 975–983,
Los Angeles, California, June 2010.

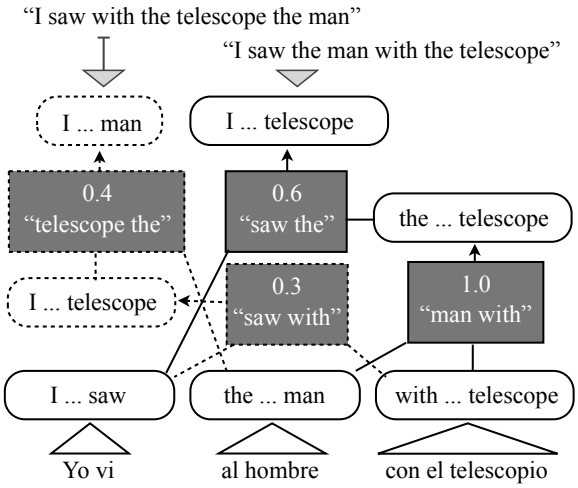

Figure 1: An example translation forest encoding two synchronous derivations for a Spanish sentence: one solid and one dotted. Nodes are annotated with their left and right unigram contexts, and hyperedges are annotated with scores $\theta \cdot \phi(r)$ and the bigrams they introduce.

best single system max-derivation baseline, and consistent improvements over a more complex multi-system pipeline that includes independent consensus decoding and system combination.

2 Model Combination

Model combination is a model-based approach to selecting translations using information from multiple component systems. Each system provides its posterior distributions over derivations $P_i(d|f)$, encoded as a weighted translation forest (i.e., translation hypergraph) in which hyperedges correspond to translation rule applications r.[2] The conditional distribution over derivations takes the form:

$$P_i(d|f) = \frac{\exp\left[\sum_{r \in d} \theta_i \cdot \phi_i(r)\right]}{\sum_{d' \in \mathcal{D}(f)} \exp\left[\sum_{r \in d'} \theta_i \cdot \phi_i(r)\right]}$$

where $\mathcal{D}(f)$ is the set of synchronous derivations encoded in the forest, r iterates over rule applications in d, and θ_i is the parameter vector for system i. The feature vector ϕ_i is system specific and includes both translation model and language model features. Figure 1 depicts an example forest.

Model combination includes four steps, described below. The entire sequence is illustrated in Figure 2.

[2] Phrase-based systems produce phrase lattices, which are instances of forests with arity 1.

2.1 Computing Combination Features

The first step in model combination is to compute n-gram expectations from component system posteriors—the same quantities found in MBR, consensus, and variational decoding techniques. For an n-gram g and system i, the expectation

$$v_i^n(g) = \mathbb{E}_{P_i(d|f)}\left[h(d, g)\right]$$

can be either an n-gram expected count, if $h(d, g)$ is the count of g in d, or the posterior probability that d contains g, if $h(d, g)$ is an indicator function. Section 3 describes how to compute these features efficiently.

2.2 Constructing a Search Space

The second step in model combination constructs a hypothesis space of translation derivations, which includes all derivations present in the forests contributed by each component system. This search space D is also a translation forest, and consists of the conjoined union of the component forests. Let R_i be the root node of component hypergraph D_i. For all i, we include all of D_i in D, along with an edge from R_i to R, the root of D. D may contain derivations from different types of translation systems. However, D only contains derivations (and therefore translations) that appeared in the hypothesis space of some component system. We do not intermingle the component search spaces in any way.

2.3 Features for the Combination Model

The third step defines a new combination model over all of the derivations in the search space D, and then annotates D with features that allow for efficient model inference. We use a linear model over four types of feature functions of a derivation:

1. Combination feature functions on n-grams $v_i^n(d) = \sum_{g \in \text{Ngrams}(d)} v_i^n(g)$ score a derivation according to the n-grams it contains.
2. Model score feature function b gives the model score $\theta_i \cdot \phi_i(d)$ of a derivation d under the system i that d is from.
3. A length feature ℓ computes the word length of the target-side yield of a derivation.
4. A system indicator feature α_i is 1 if the derivation came from system i, and 0 otherwise.

All of these features are local to rule applications (hyperedges) in D. The combination features provide information sharing across the derivations of different systems, but are functions of n-grams, and so can be scored on any translation forest. Model score features are already local to rule applications. The length feature is scored in the standard way. System indicator features are scored only on the hyperedges from R_i to R that link each component forest to the common root.

Scoring the joint search space D with these features involves annotating each rule application r (i.e. hyperedge) with the value of each feature.

2.4 Model Training and Inference

We have defined the following combination model $s_w(d)$ with weights w over derivations d from I different component models:

$$\sum_{i=1}^{I}\left[\sum_{n=1}^{4} w_i^n v_i^n(d) + w_i^\alpha \alpha_i(d)\right] + w^b \cdot b(d) + w^\ell \cdot \ell(d)$$

Because we have assessed all of these features on local rule applications, we can find the highest scoring derivation $d^* = \arg\max_{d \in D} s_w(d)$ using standard max-sum (Viterbi) inference over D.

We learn the weights of this consensus model using hypergraph-based minimum-error-rate training (Kumar et al., 2009). This procedure maximizes the translation quality of d^* on a held-out set, according to a corpus-level evaluation metric $B(\cdot; \mathbf{e})$ that compares to a reference set $\mathbf{e}$. We used BLEU, choosing w to maximize the BLEU score of the set of translations predicted by the combination model.

3 Computing Combination Features

The combination features $v_i^n(d)$ score derivations from each model with the n-gram predictions of the others. These predictions sum over all derivations under a single component model to compute a posterior belief about each n-gram. In this paper, we compare two kinds of combination features, posterior probabilities and expected counts.[3]

[3]The model combination framework could incorporate arbitrary features on the common output space of the models, but we focus on features that have previously proven useful for consensus decoding.

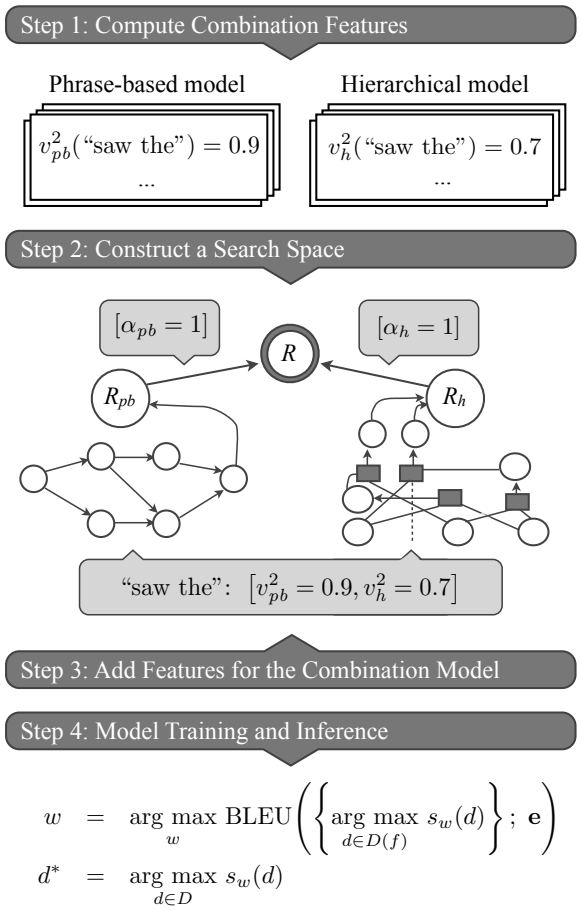

Figure 2: Model combination applied to a phrase-based (*pb*) and a hierarchical model (h) includes four steps. (1) shows an excerpt of the bigram feature function for each component, (2) depicts the result of conjoining a phrase lattice with a hierarchical forest, (3) shows example hyperedge features of the combination model, including bigram features v_i^n and system indicators α_i, and (4) gives training and decoding objectives.

Posterior probabilities represent a model's belief that the translation will contain a particular n-gram at least once. They can be expressed as $\mathbb{E}_{P(d|f)}[\delta(d, g)]$ for an indicator function $\delta(d, g)$ that is 1 if n-gram g appears in derivation d. These quantities arise in approximating BLEU for lattice-based and hypergraph-based minimum Bayes risk decoding (Tromble et al., 2008; Kumar et al., 2009). Expected n-gram counts $\mathbb{E}_{P(d|f)}[c(d, g)]$ represent the model's belief of how many times an n-gram g will appear in the translation. These quantities appear in forest-based consensus decoding (DeNero et al., 2009) and variational decoding (Li et al., 2009b).

Methods for computing both of these quantities appear in the literature. However, we address two outstanding issues below. In Section 5, we also compare the two quantities experimentally.

3.1 Computing N-gram Posteriors Exactly

Kumar et al. (2009) describes an efficient *approximate* algorithm for computing n-gram posterior probabilities. Algorithm 1 is an *exact* algorithm that computes all n-gram posteriors from a forest in a single inside pass. The algorithm tracks two quantities at each node n: regular inside scores $\beta(n)$ and n-gram inside scores $\hat{\beta}(n, g)$ that sum the scores of all derivations rooted at n that contain n-gram g.

For each hyperedge, we compute $\bar{b}(g)$, the sum of scores for derivations that *do not* contain g (Lines 8-11). We then use that quantity to compute the score of derivations that *do* contain g (Line 17).

Algorithm 1 Computing n-gram posteriors

1: **for** $n \in N$ in topological order **do**
2: $\quad\beta(n) \leftarrow 0$
3: $\quad\hat{\beta}(n, g) \leftarrow 0,\ \forall g \in \text{Ngrams}(n)$
4: $\quad$**for** $r \in \text{Rules}(n)$ **do**
5: $\quad\quad w \leftarrow \exp\left[\theta \cdot \phi(r)\right]$
6: $\quad\quad b \leftarrow w$
7: $\quad\quad\bar{b}(g) \leftarrow w,\ \forall g \in \text{Ngrams}(n)$
8: $\quad\quad$**for** $\ell \in \text{Leaves}(r)$ **do**
9: $\quad\quad\quad b \leftarrow b \times \beta(\ell)$
10: $\quad\quad\quad$**for** $g \in \text{Ngrams}(n)$ **do**
11: $\quad\quad\quad\quad\bar{b}(g) \leftarrow \bar{b}(g) \times \left(\beta(\ell) - \hat{\beta}(\ell, g)\right)$
12: $\quad\quad\beta(n) \leftarrow \beta(n) + b$
13: $\quad\quad$**for** $g \in \text{Ngrams}(n)$ **do**
14: $\quad\quad\quad$**if** $g \in \text{Ngrams}(r)$ **then**
15: $\quad\quad\quad\quad\hat{\beta}(n, g) \leftarrow \hat{\beta}(n, g) + b$
16: $\quad\quad\quad$**else**
17: $\quad\quad\quad\quad\hat{\beta}(n, g) \leftarrow \hat{\beta}(n, g) + b - \bar{b}(g)$
18: **for** $g \in \text{Ngrams}(\text{root})$ (all g in the HG) **do**
19: $\quad P(g|f) \leftarrow \frac{\hat{\beta}(\text{root}, g)}{\beta(\text{root})}$

This algorithm can in principle compute the posterior probability of any indicator function on local features of a derivation. More generally, this algorithm demonstrates how vector-backed inside passes can compute quantities beyond expectations of local features (Li and Eisner, 2009).[4] Chelba and Mahajan (2009) developed a similar algorithm for lattices.

[4]Indicator functions on derivations are not locally additive over the rules of a derivation, even if the features they indicate are local. Therefore, Algorithm 1 is not an instance of an expectation semiring computation.

3.2 Ensuring N-gram Locality

DeNero et al. (2009) describes an efficient algorithm for computing n-gram expected counts from a translation forest. This method assumes *n-gram locality* of the forest, the property that any n-gram introduced by a hyperedge appears in *all* derivations that include the hyperedge. However, decoders may recombine forest nodes whenever the language model does not distinguish between n-grams due to backoff (Li and Khudanpur, 2008). In this case, a forest encoding of a posterior distribution may not exhibit n-gram locality in all regions of the search space. Figure 3 shows a hypergraph which contains nonlocal trigrams, along with its local expansion.

Algorithm 2 expands a forest to ensure n-gram locality while preserving the encoded distribution over derivations. Let a forest (N, R) consist of nodes N and hyperedges R, which correspond to rule applications. Let $\text{Rules}(n)$ be the subset of R rooted by n, and $\text{Leaves}(r)$ be the leaf nodes of rule application r. The expanded forest (N_e, R_e) is constructed by a function $\text{Reapply}(r, L)$ that applies the rule of r to a new set of leaves $L \subset N_e$, forming a pair (r', n') consisting of a new rule application r' rooted by n'. P is a map from nodes in N to subsets of N_e which tracks how N projects to N_e. Two nodes in N_e are identical if they have the same $(n-1)$-gram left and right contexts and are projections of the same node in N. The symbol $\bigotimes$ denotes a set cross-product.

Algorithm 2 Expanding for n-gram locality

1: $N_e \leftarrow \{\}$; $R_e \leftarrow \{\}$
2: **for** $n \in N$ in topological order **do**
3: $\quad P(n) \leftarrow \{\}$
4: $\quad$**for** $r \in \text{Rules}(n)$ **do**
5: $\quad\quad$**for** $L \in \bigotimes_{\ell \in \text{Leaves}(r)} [P(\ell)]$ **do**
6: $\quad\quad\quad r', n' \leftarrow \text{Reapply}(r, L)$
7: $\quad\quad\quad P(n) \leftarrow P(n) \cup \{n'\}$
8: $\quad\quad\quad N_e \leftarrow N_e \cup \{n'\}$
9: $\quad\quad\quad R_e \leftarrow R_e \cup \{r'\}$

This transformation preserves the original distribution over derivations by splitting states, but maintaining continuations from those split states by duplicating rule applications. The process is analogous

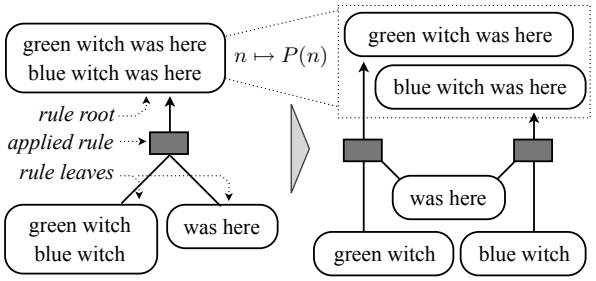

Figure 3: Hypergraph expansion ensures n-gram locality without affecting the distribution over derivations. In the left example, trigrams "green witch was" and "blue witch was" are non-local due to language model back-off. On the right, states are split to enforce trigram locality.

to expanding bigram lattices to encode a trigram history at each lattice node (Weng et al., 1998).

4 Relationship to Prior Work

Model combination is a multi-system generalization of consensus or minimum Bayes risk decoding. When only one component system is included, model combination is identical to minimum Bayes risk decoding over hypergraphs, as described in Kumar et al. (2009).[5]

4.1 System Combination

System combination techniques in machine translation take as input the outputs $\{e_1, \cdots, e_k\}$ of k translation systems, where e_i is a structured translation object (or k-best lists thereof), typically viewed as a sequence of words. The dominant approach in the field chooses a primary translation e_p as a *backbone*, then finds an alignment a_i to the backbone for each e_i. A new search space is constructed from these backbone-aligned outputs, and then a voting procedure or feature-based model predicts a final consensus translation (Rosti et al., 2007). Model combination entirely avoids this alignment problem by viewing hypotheses as n-gram occurrence vectors rather than word sequences.

Model combination also requires less total computation than applying system combination to consensus-decoded outputs. The best consensus decoding methods for individual systems already require the computation-intensive steps of model combination: producing lattices or forests, computing n-gram feature expectations, and re-decoding to maximize a secondary consensus objective. Hence, to maximize the performance of system combination, these steps must be performed for *each* system, whereas model combination requires only one forest rescoring pass over all systems.

Model combination also leverages aggregate statistics from the components' posteriors, whereas system combiners typically do not. Zhao and He (2009) showed that n-gram posterior features are useful in the context of a system combination model, even when computed from k-best lists.

Despite these advantages, system combination may be more appropriate in some settings. In particular, model combination is designed primarily for statistical systems that generate hypergraph outputs. Model combination can in principle integrate a non-statistical system that generates either a single hypothesis or an unweighted forest.[6] Likewise, the procedure could be applied to statistical systems that only generate k-best lists. However, we would not expect the same strong performance from model combination in these constrained settings.

4.2 Joint Decoding and Collaborative Decoding

Liu et al. (2009) describes two techniques for combining multiple synchronous grammars, which the authors characterize as joint decoding. Joint decoding does not involve a consensus or minimum-Bayes-risk decoding objective; indeed, their best results come from standard max-derivation decoding (with a multi-system grammar). More importantly, their computations rely on a correspondence between nodes in the hypergraph outputs of different systems, and so they can only joint decode over models with similar search strategies. We combine a phrase-based model that uses left-to-right decoding with two hierarchical systems that use bottom-up decoding — a scenario to which joint decoding is not applicable. Though Liu et al. (2009) rightly point out that most models can be decoded either left-to-

[5] We do not refer to model combination as a minimum Bayes risk decoding procedure despite this similarity because *risk* implies a belief distribution over outputs, and we now have multiple output distributions that are not necessarily calibrated. Moreover, our generalized, multi-model objective (Section 2.4) is motivated by BLEU, but not a direct approximation to it.

[6] A single hypothesis can be represented as a forest, while an unweighted forest could be assigned a uniform distribution.

right or bottom-up, such changes can have substantial implications for search efficiency and search error. We prefer to maintain the flexibility of using different search strategies in each component system.

Li et al. (2009a) is another related technique for combining translation systems by leveraging model predictions of n-gram features. K-best lists of partial translations are iteratively reranked using n-gram features from the predictions of other models (which are also iteratively updated). Our technique differs in that we use no k-best approximations, have fewer parameters to learn (one consensus weight vector rather than one for each collaborating decoder) and produce only one output, avoiding an additional system combination step at the end.

5 Experiments

We report results on the constrained data track of the NIST 2008 Arabic-to-English (ar-en) and Chinese-to-English (zh-en) translation tasks.[7] We train on all parallel and monolingual data allowed in the track. We use the NIST 2004 eval set (*dev*) for optimizing parameters in model combination and test on the NIST 2008 evaluation set. We report results using the IBM implementation of the BLEU score which computes the brevity penalty using the closest reference translation for each segment (Papineni et al., 2002). We measure statistical significance using 95% confidence intervals computed using paired bootstrap resampling. In all table cells (except for Table 3) systems without statistically significant differences are marked with the same superscript.

5.1 Base Systems

We combine outputs from three systems. Our phrase-based system is similar to the alignment template system described by Och and Ney (2004). Translation is performed using a standard left-to-right beam-search decoder. Our hierarchical systems consist of a syntax-augmented system (SAMT) that includes target-language syntactic categories (Zollmann and Venugopal, 2006) and a Hiero-style system with a single non-terminal (Chiang, 2007). Each base system yields state-of-the-art translation performance, summarized in Table 1.

[7] http://www.nist.gov/speech/tests/mt

		BLEU (%)			
		ar-en		**zh-en**	
Sys	**Base**	**dev**	**nist08**	**dev**	**nist08**
PB	MAX	51.6	43.9	37.7	25.4
PB	MBR	52.4*	44.6*	38.6*	27.3*
PB	CON	52.4*	44.6*	38.7*	27.2*
Hiero	MAX	50.9	43.3	40.0	27.2
Hiero	MBR	51.4*	43.8*	40.6*	27.8
Hiero	CON	51.5*	43.8*	40.5*	28.2
SAMT	MAX	51.7	43.8	40.8*	28.4
SAMT	MBR	52.7*	44.5*	41.1*	28.8*
SAMT	CON	52.6*	44.4*	41.1*	28.7*

Table 1: Performance of baseline systems.

	BLEU (%)			
	ar-en		**zh-en**	
Approach	**dev**	**nist08**	**dev**	**nist08**
Best MAX system	51.7	43.9	40.8	28.4
Best MBR system	52.7	44.5	41.1	28.8*
MC Conjoin/SI	**53.5**	**45.3**	**41.6**	**29.0***

Table 2: Performance from the best single system for each language pair without consensus decoding (*Best MAX system*), the best system with minimum Bayes risk decoding (*Best MBR system*), and model combination across three systems.

For each system, we report the performance of max-derivation decoding (MAX), hypergraph-based MBR (Kumar et al., 2009), and a linear version of forest-based consensus decoding (CON) (DeNero et al., 2009). MBR and CON differ only in that the first uses n-gram posteriors, while the second uses expected n-gram counts. The two consensus decoding approaches yield comparable performance. Hence, we report performance for hypergraph-based MBR in our comparison to model combination below.

5.2 Experimental Results

Table 2 compares model combination (MC) to the best MAX and MBR systems. Model combination uses a conjoined search space wherein each hyperedge is annotated with 21 features: 12 n-gram posterior features v_i^n computed from the PB/Hiero/SAMT forests for $n \leq 4$; 4 n-gram posterior features v^n computed from the conjoined forest; 1 length feature ℓ; 1 feature b for the score assigned by the base model; and 3 system indicator (SI) features α_i that select which base system a derivation came from. We refer to this model combination approach as MC

Strategy	BLEU (%) ar-en dev	ar-en nist08	zh-en dev	zh-en nist08
Best MBR system	52.7	44.5	41.1	28.8
MBR Conjoin	52.3	44.5	40.5	28.3
MBR Conjoin/feats-best	52.7	44.9	41.2	28.8
MBR Conjoin/SI	53.1	44.9	41.2	28.9
MC 1-best HG	52.7	44.6	41.1	28.7
MC Conjoin	52.9	44.6	40.3	28.1
MC Conjoin/base/SI	53.5	45.1	41.2	28.9
MC Conjoin/SI	**53.5**	**45.3**	**41.6**	**29.0**

Table 3: Model Combination experiments.

Approach	Base	BLEU (%) ar-en dev	ar-en nist08	zh-en dev	zh-en nist08
Sent-level	MAX	51.8^{*}	44.4^{*}	40.8^{*}	28.2^{*}
Word-level	MAX	52.0^{*}	44.4^{*}	40.8^{*}	28.1^{*}
Sent-level	MBR	52.7^{+}	44.6^{*}	41.2	28.8^{+}
Word-level	MBR	52.5^{+}	44.7^{*}	40.9	28.8^{+}
MC-conjoin-SI		**53.5**	**45.3**	**41.6**	**29.0**$^{+}$

Table 4: BLEU performance for different system and model combination approaches. Sentence-level and word-level system combination operate over the sentence output of the base systems, which are either decoded to maximize derivation score (MAX) or to minimize Bayes risk (MBR).

Conjoin/SI. Model combination improves over the single best MAX system by 1.4 BLEU in ar-en and 0.6 BLEU in zh-en, and always improves over MBR.

This improvement could arise due to multiple reasons: a bigger search space, the consensus features from constituent systems, or the system indicator features. Table 3 teases apart these contributions.

We first perform MBR on the conjoined hypergraph (MBR-Conjoin). In this case, each edge is tagged with 4 conjoined n-gram features v^n, along with length and base model features. MBR-Conjoin is worse than MBR on the hypergraph from the single best system. This could imply that either the larger search space introduces poor hypotheses or that the n-gram posteriors obtained are weaker. When we now restrict the n-gram features to those from the best system (MBR Conjoin/feats-best), BLEU scores increase relative to MBR-Conjoin. This implies that the n-gram features computed over the conjoined hypergraph are weaker than the corresponding features from the best system.

Adding system indicator features (MBR Conjoin+SI) helps the MBR-Conjoin system considerably; the resulting system is better than the best MBR system. This could mean that the SI features guide search towards stronger parts of the larger search space. In addition, these features provide a normalization of scores across systems.

We next do several model-combination experiments. We perform model combination using the search space of only the best MBR system (MC 1best HG). Here, the hypergraph is annotated with n-gram features from the 3 base systems, as well as length and base model features. A total of $3 \times 4 + 1 + 1 = 14$ features are added to each edge. Surprisingly, n-gram features from the additional systems did not help select a better hypothesis within the search space of a single system.

When we expand the search space to the conjoined hypergraph (MC Conjoin), it performs worse relative to MC 1-best. Since these two systems are identical in their feature set, we hypothesize that the larger search space has introduced erroneous hypotheses. This is similar to the scenario where MBR Conjoin is worse than MBR 1-best. As in the MBR case, adding system indicator features helps (MC Conjoin/base/SI). The result is comparable to MBR on the conjoined hypergraph with SI features.

We finally add extra n-gram features which are computed from the conjoined hypergraph (MC Conjoin + SI). This gives the best performance although the gains over MC Conjoin/base/SI are quite small. Note that these added features are the same n-gram features used in MBR Conjoin. Although they are not strong by themselves, they provide additional discriminative power by providing a consensus score across all 3 base systems.

5.3 Comparison to System Combination

Table 4 compares model combination to two system combination algorithms. The first, which we call *sentence-level* combination, chooses among the base systems' three translations the sentence that has the highest consensus score. The second, *word-level* combination, builds a "word sausage" from the outputs of the three systems and chooses a path through the sausage with the highest score under a similar model (Macherey and Och, 2007). Nei-

	BLEU (%)			
	ar-en		zh-en	
Approach	**dev**	**nist08**	**dev**	**nist08**
HG-expand	52.7*	44.5*	41.1*	28.8*
HG-noexpand	52.7*	44.5*	41.1*	28.8*

Table 5: MBR decoding on the syntax augmented system, with and without hypergraph expansion.

	BLEU (%)			
	ar-en		zh-en	
Posteriors	**dev**	**nist08**	**dev**	**nist08**
Exact	52.4*	44.6*	38.6*	27.3*
Approximate	52.5*	44.6*	38.6*	27.2*

Table 6: MBR decoding on the phrase-based system with either exact or approximate posteriors.

ther system combination technique provides much benefit, presumably because the underlying systems all share the same data, pre-processing, language model, alignments, and code base.

Comparing system combination when no consensus (i.e., minimum Bayes risk) decoding is utilized at all, we find that model combination improves upon the result by up to 1.1 BLEU points. Model combination also performs slightly better relative to system combination over MBR-decoded systems. In the latter case, system combination actually requires more computation compared to model combination; consensus decoding is performed for *each* system rather than only once for model combination. This experiment validates our approach. Model combination outperforms system combination while avoiding the challenge of aligning translation hypotheses.

5.4 Algorithmic Improvements

Section 3 describes two improvements to computing n-gram posteriors: hypergraph expansion for n-gram locality and exact posterior computation. Table 5 shows MBR decoding with and without expansion (Algorithm 2) in a decoder that collapses nodes due to language model back-off. These results show that while expansion is necessary for correctness, it does not affect performance.

Table 6 compares exact n-gram posterior computation (Algorithm 1) to the approximation described by Kumar et al. (2009). Both methods yield identical results. Again, while the exact method guarantees correctness of the computation, the approximation suffices in practice.

6 Conclusion

Model combination is a consensus decoding strategy over a collection of forests produced by multiple machine translation systems. These systems can have varied decoding strategies; we only require that each system produce a forest (or a lattice) of translations. This flexibility allows the technique to be applied quite broadly. For instance, de Gispert et al. (2009) describe combining systems based on multiple source representations using minimum Bayes risk decoding—likewise, they could be combined via model combination.

Model combination has two significant advantages over current approaches to system combination. First, it does not rely on hypothesis alignment between outputs of individual systems. Aligning translation hypotheses accurately can be challenging, and has a substantial effect on combination performance (He et al., 2008). Instead of aligning hypotheses, we compute expectations of local features of n-grams. This is analogous to how BLEU score is computed, which also views sentences as vectors of n-gram counts (Papineni et al., 2002) . Second, we do not need to pick a backbone system for combination. Choosing a backbone system can also be challenging, and also affects system combination performance (He and Toutanova, 2009). Model combination sidesteps this issue by working with the conjoined forest produced by the union of the component forests, and allows the consensus model to express system preferences via weights on system indicator features.

Despite its simplicity, model combination provides strong performance by leveraging existing consensus, search, and training techniques. The technique outperforms MBR and consensus decoding on each of the component systems. In addition, it performs better than standard sentence-based or word-based system combination techniques applied to either max-derivation or MBR outputs of the individual systems. In sum, it is a natural and effective model-based approach to multi-system decoding.

References

Ciprian Chelba and M. Mahajan. 2009. A dynamic programming algorithm for computing the posterior probability of n-gram occurrences in automatic speech recognition lattices. Personal communication.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*.

A. de Gispert, S. Virpioja, M. Kurimo, and W. Byrne. 2009. Minimum bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

John DeNero, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of the Association for Computational Linguistics and IJCNLP*.

Robert Frederking and Sergei Nirenburg. 1994. Three heads are better than one. In *Proceedings of the Conference on Applied Natural Language Processing*.

Xiaodong He and Kristina Toutanova. 2009. Joint optimization for machine translation system combination. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Xiaodong He, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. 2008. Indirect-hmm-based hypothesis alignment for combining outputs from machine translation systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Association for Computational Linguistics and IJCNLP*.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Zhifei Li and Sanjeev Khudanpur. 2008. A scalable decoder for parsing-based machine translation with equivalent language model state maintenance. In *ACL Workshop on Syntax and Structure in Statistical Translation*.

Mu Li, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009a. Collaborative decoding: Partial hypothesis re-ranking using translation consensus between decoders. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Zhifei Li, Jason Eisner, and Sanjeev Khudanpur. 2009b. Variational decoding for statistical machine translation. In *Proceedings of the Association for Computational Linguistics and IJCNLP*.

Yang Liu, Haitao Mi, Yang Feng, and Qun Liu. 2009. Joint decoding with multiple translation models. In *Proceedings of the Association for Computational Linguistics and IJCNLP*.

Wolfgang Macherey and Franz Och. 2007. An empirical study on computing consensus translations from multiple machine translation systems. In *EMNLP*, Prague, Czech Republic.

Franz J. Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417 – 449.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*.

Antti-Veikko I. Rosti, Necip Fazil Ayan, Bing Xiang, Spyros Matsoukas, Richard Schwartz, and Bonnie J. Dorr. 2007. Combining outputs from multiple machine translation systems. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Roy Tromble, Shankar Kumar, Franz J. Och, and Wolfgang Macherey. 2008. Lattice minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Fuliang Weng, Andreas Stolcke, and Ananth Sankar. 1998. Efficient lattice representation and generation. In *Intl. Conf. on Spoken Language Processing*.

Yong Zhao and Xiaodong He. 2009. Using n-gram based features for machine translation system combination. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of the NAACL 2006 Workshop on statistical machine translation*.