BBN System Description for WMT10 System Combination Task

Antti-Veikko I. Rosti and **Bing Zhang** and **Spyros Matsoukas** and **Richard Schwartz**
Raytheon BBN Technologies, 10 Moulton Street, Cambridge, MA 02138, USA
{arosti,bzhang,smatsouk,schwartz}@bbn.com

Abstract

BBN submitted system combination outputs for Czech-English, German-English, Spanish-English, French-English, and All-English language pairs. All combinations were based on confusion network decoding. An incremental hypothesis alignment algorithm with flexible matching was used to build the networks. The bi-gram decoding weights for the single source language translations were tuned directly to maximize the BLEU score of the decoding output. Approximate expected BLEU was used as the objective function in gradient based optimization of the combination weights for a 44 system multi-source language combination (All-English). The system combination gained around 0.4-2.0 BLEU points over the best individual systems on the single source conditions. On the multi-source condition, the system combination gained 6.6 BLEU points.

1 Introduction

The BBN submissions to the WMT10 system combination task were based on confusion network decoding. The confusion networks were built using the incremental hypothesis alignment algorithm with flexible matching introduced in the BBN submission for the WMT09 system combination task (Rosti et al., 2009). This year, the system combination weights were tuned to maximize the BLEU score (Papineni et al., 2002) of the 1-best decoding output (lattice based BLEU tuning) using downhill simplex method (Press et al., 2007). A 44 system multi-source combination was also submitted. Since the gradient-free optimization algorithms do not seem to be able to handle more than 20-30 weights, a gradient ascent to maximize an approximate expected BLEU objective was used to optimize the larger number of weights.

The lattice based BLEU tuning may be implemented using any optimization algorithm that does not require the gradient of the objective function. Due to the size of the lattices, the objective function evaluation may have to be distributed to multiple servers. The optimizer client accumulates the BLEU statistics of the 1-best hypotheses from the servers for given search weights, computes the final BLEU score, and passes it to the optimization algorithm which returns a new set of search weights. The lattice based tuning explores the entire search space and does not require multiple decoding iterations with N-best list merging to approximate the search space as in the standard minimum error rate training (Och, 2003). This allows much faster turnaround in weight tuning.

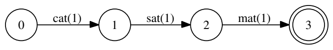
Differentiable approximations of BLEU have been proposed for consensus decoding. Tromble et al. (2008) used a linear approximation and Pauls et al. (2009) used a closer approximation called CoBLEU. CoBLEU is based on the BLEU formula but the n-gram counts are replaced by expected counts over a translation forest. Due to the min-functions required in converting the n-gram counts to matches and a non-differentiable brevity penalty, a sub-gradient ascent must be used. In this work, an approximate expected BLEU (ExpBLEU) defined over N-best lists was used as a differentiable objective function. ExpBLEU uses expected BLEU statistics where the min-function is not needed as the statistics are computed offline and the brevity penalty is replaced by a differentiable approximation. The ExpBLEU tuning yields comparable results to direct BLEU tuning using gradient-free algorithms on combinations of small number of systems (fewer than 20-30 weights). Results on a 44 system combination show that the gradient based optimization is more robust with larger number of weights.

This paper is organized as follows. Section 2 reviews the incremental hypothesis alignment algorithm used to built the confusion networks. Decoding weight optimization using direct lattice 1-best BLEU tuning and N-best list based ExpBLEU tuning are presented in Section 3. Experimental results on combining single source language to English outputs and all 44 English outputs are detailed in Section 4. Finally, Section 5 concludes this paper with some ideas for future work.

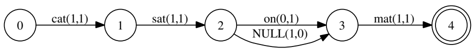
2 Hypothesis Alignment

The confusion networks were built by using the incremental hypothesis alignment algorithm with flexible matching introduced in Rosti et al. (2009). The algorithm is reviewed in more detail here. It is loosely related to the alignment performed in the calculation of the translation edit rate (TER) (Snover et al., 2006) which estimates the edit distance between two strings allowing shifts of blocks of words in addition to insertions, deletions, and substitutions. Calculating an exact TER for strings longer than a few tokens[1] is not computationally feasible, so the `tercom`[2] software uses heuristic shift constraints and pruning to find an upper bound of TER. In this work, the hypotheses were aligned incrementally with the confusion network, thus using tokens from all previously aligned hypotheses in computing the edit distance. Lower substitution costs were assigned to tokens considered equivalent and the heuristic shift constraints of `tercom` were relaxed[3].

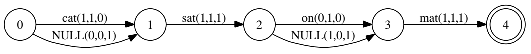
First, tokens from all hypotheses are put into equivalence classes if they belong to the same WordNet (Fellbaum, 1998) synonym set or have the same stem. The 1-best hypothesis from each system is used as the confusion network skeleton which defines the final word order of the decoding output. Second, a trivial confusion network is generated from the skeleton hypothesis by generating a single arc for each token. The alignment algorithm explores shifts of blocks of words that minimize the edit distance between the current confusion network and an unaligned hypothesis. Third, the hypothesis with the lowest edit distance to the current confusion network is aligned into the network. The heuristically selected edit costs used in the WMT10 system were 1.0 for insertions, deletions, and shifts, 0.2 for substitutions of tokens in the same equivalence class, and 1.0001 for substitutions of non-equivalent tokens. An insertion with respect to the network always results in a new node and two new arcs. The first arc contains the inserted token and the second arc contains a NULL token representing the missing token from all previously aligned hypotheses. A substitution/deletion results in a new token/NULL arc or increase in the confidence of an existing token/NULL arc. The process is repeated until all hypotheses are aligned into the network.



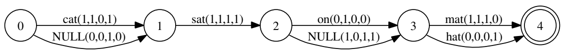
(a) Skeleton hypothesis.



(b) Two hypotheses (insertion).



(c) Three hypotheses (deletion).



(d) Four hypotheses (substitution).

Figure 1: Example of incrementally aligning “cat sat mat”, “cat sat on mat”, “sat mat”, and “cat sat hat”.

For example, given the following hypotheses from four systems: “cat sat mat”, “cat sat on mat”, “sat mat”, and “cat sat hat”, an initial network in Figure 1(a) is generated. The following two hypotheses have a distance of one edit from the initial network, so the second can be aligned next. Figure 1(b) shows the additional node created and the two new arcs for ‘on’ and ‘NULL’ tokens. The third hypothesis has deleted token ‘cat’ and matches the

[1]Hypotheses are tokenized and lower-cased prior to alignment. Tokens generally refer to words and punctuation.

[2]`http://www.cs.umd.edu/˜snover/tercom/` current version 0.7.25.

[3]This algorithm is not equivalent to an incremental TER-Plus (Snover et al., 2009) due to different shift constraints and the lack of paraphrase matching

'NULL' token between nodes 2 and 3 as seen in Figure 1(c). The fourth hypothesis matches all but the final token 'hat' which becomes a substitution for 'mat' in Figure 1(d). The binary vectors in the parentheses following each token show which system generated the token aligned to that arc. If the systems generated N-best hypotheses, a fractional increment could be added to these vectors as in (Rosti et al., 2007). Given these system specific scores are normalized to sum to one over all arcs connecting two consecutive nodes, they may be viewed as system specific word arc posterior estimates. Note, for 1-best hypotheses the scores sum to one without normalization.

Given system outputs $\mathcal{E} = \{E_1, \ldots, E_{N_s}\}$, an algorithm to build a set of N_s confusion networks $\mathcal{C} = \{C_1, \ldots, C_{N_s}\}$ may be written as:

```
**for** $n = 1$ to $N_s$ **do**
    $C_n \Leftarrow \mathrm{Init}(E_n)$ {initialize confusion network from the skeleton}
    $\mathcal{E}' \Leftarrow \mathcal{E} - E_n$ {set of unaligned hypotheses}
    **while** $\mathcal{E}' \neq \emptyset$ **do**
        $E_m \Leftarrow \arg\min_{E \in \mathcal{E}'} \mathrm{Dist}(E, C_n)$ {compute edit distances}
        $C_n \Leftarrow \mathrm{Align}(E_m, C_n)$ {align closest hypothesis}
        $\mathcal{E}' \Leftarrow \mathcal{E}' - E_m$ {update set of unaligned hypotheses}
    **end while**
**end for**
```

The set of N_s confusion networks are expanded to separate paths with distinct bi-gram contexts and connected in parallel into a big lattice with common start and end nodes with NULL token arcs. A prior probability estimate is assigned to the system specific word arc confidences connecting the common start node and the first node in each subnetwork. A heuristic prior is estimated as:

$$p_n = \frac{1}{Z} \exp(-100 \frac{e_n}{N_n}) \qquad (1)$$

where e_n is the total cost of aligning all hypotheses when using system n as the skeleton, N_n is the number of nodes in the confusion network before bi-gram expansion, and Z is a scaling factor to guarantee p_n sum to one. This gives a higher prior for a network with fewer alignment errors and longer expected decoding output.

3 Weight Optimization

Standard search algorithms may be used to find N-best hypotheses from the final lattice. The score for arc l is computed as:

$$s_l = \log\Big(\sum_{n=1}^{N_s} \sigma_n s_{nl}\Big) + \lambda L(w_l|w_{P(l)}) + \omega S(w_l) \qquad (2)$$

where σ_n are the system weights constrained to sum to one, s_{nl} are the system specific arc posteriors, λ is a language model (LM) scaling factor, $L(w_l|w_{P(l)})$ is the bi-gram log-probability for the token w_l on the arc l given the token $w_{P(l)}$ on the arc $P(l)$ preceding the arc l, ω is the word insertion scaling factor, and $S(w_l)$ is zero if w_l is a NULL token and one otherwise. The path with the highest total score under summation is the 1-best decoding output. The decoding weights $\theta = \{\sigma_1, \ldots, \sigma_{N_s}, \lambda, \omega\}$ are tuned to optimize two objective functions described next.

3.1 Lattice Based BLEU Optimization

Powell's method (Press et al., 2007) on N-best lists was used in system combination weight tuning in Rosti et al. (2007). This requires multiple decoding iterations and merging the N-best lists between tuning runs to approximate the full search space as in Och (2003). To speed up the tuning process, a distributed optimization method can be used. The lattices are divided into multiple chunks each of which are loaded into memory by a server. A client runs the optimization algorithm relying on the servers for parallelized objective function evaluation. The client sends a new set of search weights to the servers which decode the chunks of lattices and return the 1-best hypothesis BLEU statistics back to the client. The client accumulates the BLEU statistics from all servers and computes the final BLEU score used as the objective function by the optimization algorithm. Results similar to Powell's method can be obtained with fewer iterations by using the downhill simplex method in multi-dimensions (Amoeba) (Press et al., 2007). To enforce the sum to one constraint of the system weights σ_n, the search weights are restricted to $[0, 1]$ by assigning a large penalty if any corresponding search weight breaches the limits and these restricted search weights are scaled to sum to one before the objective function evaluation.

After optimizing the bi-gram decoding weights directly on the lattices, a 300-best list are generated. The 300-best hypotheses are re-scored using a 5-gram LM and another set of re-scoring weights are tuned on the development set using the standard N-best list based method. Multiple random restarts may be used in both lattice and N-best list based optimization to decrease chances of finding a local minimum. Twenty sets of initial weights (the weights from the previous tuning and 19 randomly perturbed weights) were used in all experiments.

3.2 Approximate Expected BLEU Optimization

The gradient-free optimization algorithms like Powell's method and downhill simplex work well for up to around 20-30 weights. When the number of weights is larger, the algorithms often get stuck in local optima even if multiple random restarts are used. The BLEU score for a 1-best output is defined as follows:

$$\text{BLEU} = \prod_{n=1}^{4} \left(\frac{\sum_i m_i^n}{\sum_i h_i^n} \right)^{\frac{1}{4}} \phi\left(1 - \frac{\sum_i r_i}{\sum_i h_i^1}\right) \quad (3)$$

where m_i^n is the number of n-gram matches between the hypothesis and reference for segment i, h_i^n is the number of n-grams in the hypothesis, r_i is the reference length (or the reference length closest to the hypothesis if multiple references are available), and $\phi(x) = \min(1.0, e^x)$ is the brevity penalty. The first term in Equation 3 is a harmonic mean of the n-gram precisions up to $n = 4$. The selection of 1-best hypotheses is discrete and the brevity penalty is not continuous, so the BLEU score is not differentiable and gradient based optimization cannot be used. Given a posterior distribution over all possible decoding outputs could be defined, an expected BLEU could be optimized using gradient ascent. However, this posterior distribution can only be approximated by expensive sampling methods.

A differentiable objective function over N-best lists to approximate the BLEU score can be defined using expected BLEU statistics and a continuous approximation of the brevity penalty. The posterior probability for hypothesis j of segment i is simply the normalized decoder score:

$$p_{ij} = \frac{e^{\gamma S_{ij}}}{\sum_k e^{\gamma S_{ik}}} \quad (4)$$

where γ is a posterior scaling factor and S_{ij} is the total score of hypothesis j of segment i. The posterior scaling factor controls the shape of the posterior distribution: $\gamma > 1.0$ moves the probability mass toward the 1-best hypothesis and $\gamma < 1.0$ flattens the distribution. The BLEU statistics in Equation 3 are replaced by the expected statistics; for example, $\hat{m}_i^n = \sum_j p_{ij} m_{ij}$, and the brevity penalty $\phi(x)$ is approximated by:

$$\varphi(x) = \frac{e^x - 1}{e^{1000x} + 1} + 1 \quad (5)$$

ExpBLEU has a closed form solution for the gradient, provided the total decoder score is differentiable.

The penalty used to restrict the search weights corresponding to the system weights σ_n in gradient-free BLEU tuning is not differentiable. For expected BLEU tuning, the search weights ς_n are unrestricted but the system weights are obtained by a sigmoid transform and normalized to sum to one:

$$\sigma_n = \frac{\delta(\varsigma_n)}{\sum_m \delta(\varsigma_m)} \quad (6)$$

where $\delta(\varsigma_n) = 1/(1 + e^{-\varsigma_n})$.

The expected BLEU tuning is performed on N-best lists in similar fashion to direct BLEU tuning. Tuned weights from one decoding iteration are used to generate a new N-best list, the new N-best list is merged with the N-best list from the previous tuning run, and a new set of weights are optimized using limited memory Broyden-Fletcher-Goldfarb-Shanno method (lBFGS) (Liu and Nocedal, 1989). Since the posterior distribution is affected by the size of the N-best list and different decoding weights, the posterior scaling factor can be set for each tuning run so that the perplexity of the posterior distribution given the merged N-best list is constant. A target perplexity of 5.0 was used in the experiments. Four iterations of bi-gram decoding weight tuning were performed using 300-best lists. The final 300-best list was re-scored with a 5-gram and another set of re-scoring weights was tuned on the development set.

4 Experimental Evaluation

System outputs for all language pairs with English as the target were combined. Unpruned English bi-gram and 5-gram language model components were trained using the WMT10 corpora: `EuroParl`, `GigaFrEn`, `NewsCommentary`, and `News`. Additional six Gigaword v4 components were trained: `AFP`, `APW`, `XIN+CNA`,

tune	`cz-en`		`de-en`		`es-en`		`fr-en`	
System	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
worst	68.99	13.85	68.45	15.07	60.86	21.02	71.17	15.00
best	56.77	22.84	57.76	25.05	51.81	30.10	53.66	28.64
syscomb	57.31	25.11	54.97	27.75	50.46	31.54	51.35	31.16

test	`cz-en`		`de-en`		`es-en`		`fr-en`	
System	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
worst	68.65	14.29	67.50	15.66	60.52	21.86	68.36	16.82
best	56.13	23.56	58.12	24.34	51.45	30.56	52.16	29.79
syscomb	56.89	25.12	55.60	26.38	50.33	31.59	51.36	30.16

Table 1: Case insensitive TER and BLEU scores on `syscombtune` (tune) and `syscombtest` (test) for combinations of outputs from four source languages.

`LTW`, `NYT`, and `Headlines+Datelines`. Interpolation weights for the ten components were tuned so as to minimize perplexity on the `newstest2009-ref.en` development set. The LMs used modified Kneser-Ney smoothing. On the multi-source condition (`xx-en`) another LM was trained from the system outputs and interpolated with the general LM using an interpolation weight 0.3 for the LM trained on the system outputs. This LM is referred to as biasLM later. A tri-gram true casing model was trained using all available English data. This model was used to restore the case of the lower-case system combination output.

All six 1-best system outputs on `cz-en`, 16 outputs on `de-en`, 8 outputs on `es-en`, and 14 outputs on `fr-en` were combined. The lattice based BLEU tuning was used to optimize the bi-gram decoding weights and N-best list based BLEU tuning was used to optimize the 5-gram rescoring weights. Results for these single source language experiments are shown in Table 1. The gains on `syscombtune` were similar to those on `syscombtest` for all but French-English. The tuning set contained only 455 segments but appeared to be well matched with the larger (2034 segments) test set. The characteristics of the individual system outputs were probably different for the tuning and test sets on French-English translation. In our experience, optimizing system combination weights using the ExpBLEU tuning for a small number of systems yields similar results to lattice based BLEU tuning. The lattice based BLEU tuning is faster as there is no need for multiple decoding and tuning iterations. Using the biasLM on the single source combinations did not yield any gains. The output for these conditions probably did not contain enough data for biasLM training given the small tuning set and small number of systems.

`xx-en`	**tune**		**test**	
System	TER	BLEU	TER	BLEU
worst	71.17	13.85	68.65	14.29
best	51.81	30.10	51.45	30.56
lattice	43.15	35.72	43.79	35.29
expBLEU	44.07	36.91	44.35	36.62
+biasLM	43.63	**37.61**	44.50	**37.12**

Table 2: Case insensitive TER and BLEU scores on `syscombtune` (tune) and `syscombtest` (test) for `xx-en` combination. Combinations using lattice BLEU tuning, expected BLEU tuning, and after adding the system output biased LM are shown.

Finally, experiments combining all 44 1-best system outputs were performed to produce a multi-source combination output. The first experiment used the lattice based BLEU tuning and gave a 5.6 BLEU point gain on the tuning set as seen in Table 2. The ExpBLEU tuning gave an additional 1.2 point gain which suggests that the direct lattice based BLEU tuning got stuck in a local optimum. Using the system output biased LM gave an additional 0.7 point gain. The gains on the test set were similar and the best combination gave a 6.6 point gain over the best individual system.

5 Conclusions

The BBN submissions for WMT10 system combination task were described in this paper. The combination was based on confusion network decoding. The confusion networks were built using an incremental hypothesis alignment algorithm with flexible matching. The bi-gram decoding weights for the single source conditions were optimized directly to maximize the BLEU scores of the 1-best decoding outputs and the 5-gram re-scoring weights were tuned on 300-best lists. The BLEU gains over the best individual system outputs were around 1.5 points on `cz-en`, 2.0 points on `de-en`, 1.0 points on `es-en`, and 0.4 points on `fr-en`. The system combination weights on `xx-en` were tuned to maximize ExpBLEU, and a system output biased LM was used. The BLEU gain over the best individual system was 6.6 points. Future work will investigate tuning of the edit costs used in the alignment. A lattice based ExpBLEU tuning will be investigated. Also, weights for more complicated functions with additional features may be tuned using ExpBLEU.

Acknowledgments

This work was supported by DARPA/IPTO Contract No. HR0011-06-C-0022 under the GALE program.

References

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory method for large scale optimization. *Mathematical Programming*, 45(3):503–528.

Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

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

Adam Pauls, John DeNero, and DanKlein. 2009. Consensus training for consensus decoding in machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1418–1427.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 2007. *Numerical recipes: the art of scientific computing*. Cambridge University Press, 3rd edition.

Antti-Veikko I. Rosti, Spyros Matsoukas, and Richard Schwartz. 2007. Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319.

Antti-Veikko I. Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2009. Incremental hypothesis alignment with flexible matching for building confusion networks: BBN system description for WMT09 system combination task. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 61–65.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciula, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231.

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or HTER? exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268.

Roy W. Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 620–629.