In Proceedings of the 51st Annual Meeting of the Association
for Computational Linguistics (ACL 2013). Aug 4-9, 2013.
Sofia, Bulgaria. Short Paper. Poster presentation

Stacking for Statistical Machine Translation[*]

Majid Razmara and **Anoop Sarkar**
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
{razmara,anoop}@sfu.ca

Abstract

We propose the use of *stacking*, an ensemble learning technique, to the statistical machine translation (SMT) models. A diverse ensemble of weak learners is created using the same SMT engine (a hierarchical phrase-based system) by manipulating the training data and a strong model is created by combining the weak models on-the-fly. Experimental results on two language pairs and three different sizes of training data show significant improvements of up to 4 BLEU points over a conventionally trained SMT model.

1 Introduction

Ensemble-based methods have been widely used in machine learning with the aim of reducing the instability of classifiers and regressors and/or increase their bias. The idea behind ensemble learning is to combine multiple models, *weak learners*, in an attempt to produce a *strong model* with less error. It has also been successfully applied to a wide variety of tasks in NLP (Tomeh et al., 2010; Surdeanu and Manning, 2010; F. T. Martins et al., 2008; Sang, 2002) and recently has attracted attention in the statistical machine translation community in various work (Xiao et al., 2013; Song et al., 2011; Xiao et al., 2010; Lagarda and Casacuberta, 2008).

In this paper, we propose a method to adopt *stacking* (Wolpert, 1992), an ensemble learning technique, to SMT. We manipulate the full set of training data, creating k disjoint sets of *held-out* and *held-in* data sets as in k-fold cross-validation and build a model on each partition. This creates a diverse ensemble of statistical machine translation models where each member of the ensemble has different feature function values for the SMT log-linear model (Koehn, 2010). The weights of model are then tuned using minimum error rate training (Och, 2003) on the *held-out* fold to provide k weak models. We then create a strong model by stacking another meta-learner on top of weak models to combine them into a single model. The particular second-tier model we use is a model combination approach called *ensemble decoding* which combines hypotheses from the weak models on-the-fly in the decoder.

Using this approach, we take advantage of the diversity created by manipulating the training data and obtain a significant and consistent improvement over a conventionally trained SMT model with a fixed training and tuning set.

2 Ensemble Learning Methods

Two well-known instances of general framework of ensemble learning are *bagging* and *boosting*. Bagging (Breiman, 1996a) (bootstrap aggregating) takes a number of samples with replacement from a training set. The generated sample set may have 0, 1 or more instances of each original training instance. This procedure is repeated a number of times and the base learner is applied to each sample to produce a weak learner. These models are aggregated by doing a uniform voting for classification or averaging the predictions for regression. Bagging reduces the variance of the base model while leaving the bias relatively unchanged and is most useful when a small change in the training data affects the prediction of the model (i.e. the model is unstable) (Breiman, 1996a). Bagging has been recently applied to SMT (Xiao et al., 2013; Song et al., 2011)

Boosting (Schapire, 1990) constructs a strong learner by repeatedly choosing a weak learner and applying it on a re-weighted training set. In each iteration, a weak model is learned on the training data, whose instance weights are modified from the previous iteration to concentrate on examples on which the model predictions were poor. By putting more weight on the wrongly predicted examples, a diverse ensemble of weak learners is created. Boosting has also been used in SMT (Xiao et al., 2013; Xiao et al., 2010; Lagarda

[*] This research was partially supported by an NSERC, Canada (RGPIN: 264905) grant and a Google Faculty Award to the second author.

Algorithm 1: Stacking for SMT

```
Input: $\mathcal{D} = \{\langle f_j, e_j \rangle\}_{j=1}^{N}$    ▷ A parallel corpus
Input: $k$    ▷ # of folds (i.e. weak learners)
Output: STRONGMODEL $s$
$\mathcal{D}^1, \dots, \mathcal{D}^k \leftarrow$ SPLIT($\mathcal{D}$, $k$)
for $i = 1 \rightarrow k$ do
    $\mathcal{T}^i \leftarrow \mathcal{D} - \mathcal{D}^i$    ▷ Use all but current partition as training set.
    $\phi_i \leftarrow$ TRAIN($\mathcal{T}^i$)    ▷ Train feature functions.
    $\mathcal{M}_i \leftarrow$ TUNE($\phi_i$, $\mathcal{D}^i$)    ▷ Tune the model on the current partition.
end for
s $\leftarrow$ COMBINEMODELS($\mathcal{M}_1$ , . . ., $\mathcal{M}_k$) ▷ Combine all the base models to produce a strong stacked model.
```

and Casacuberta, 2008).

Stacking (or stacked generalization) (Wolpert, 1992) is another ensemble learning algorithm that uses a second-level learning algorithm on top of the base learners to reduce the bias. The first level consists of predictors $g_1, \dots, g_k$ where $g_i : \mathbb{R}^d \rightarrow \mathbb{R}$, receiving input $x \in \mathbb{R}^d$ and producing a prediction $g_i(x)$. The next level consists of a single function $h : \mathbb{R}^{d+k} \rightarrow \mathbb{R}$ that takes $\langle x, g_1(x), \dots, g_k(x) \rangle$ as input and produces an ensemble prediction $\hat{y} = h(x, g_1(x), \dots, g_k(x))$.

Two categories of ensemble learning are *homogeneous learning* and *heterogeneous learning*. In homogeneous learning, a single base learner is used, and diversity is generated by data sampling, feature sampling, randomization and parameter settings, among other strategies. In heterogeneous learning different learning algorithms are applied to the same training data to create a pool of diverse models. In this paper, we focus on homogeneous ensemble learning by manipulating the training data.

In the primary form of stacking (Wolpert, 1992), the training data is split into multiple disjoint sets of *held-out* and *held-in* data sets using k-fold cross-validation and k models are trained on the held-in partitions and run on held-out partitions. Then a meta-learner uses the predictions of all models on their held-out sets and the actual labels to learn a final model. The details of the first-layer and second-layer predictors are considered to be a "black art" (Wolpert, 1992).

Breiman (1996b) linearly combines the weak learners in the stacking framework. The weights of the base learners are learned using ridge regression: $s(x) = \sum_k \alpha_k m_k(x)$, where m_k is a base model trained on the k-th partition of the data and s is the resulting strong model created by linearly interpolating the weak learners.

Stacking (aka blending) has been used in the system that won the Netflix Prize[1], which used a multi-level stacking algorithm.

Stacking has been actively used in statistical parsing: Nivre and McDonald (2008) integrated two models for dependency parsing by letting one model learn from features generated by the other; F. T. Martins et al. (2008) further formalized the stacking algorithm and improved on Nivre and McDonald (2008); Surdeanu and Manning (2010) includes a detailed analysis of ensemble models for statistical parsing: *i)* the diversity of base parsers is more important than the complexity of the models; *ii)* unweighted voting performs as well as weighted voting; and *iii)* ensemble models that combine at decoding time significantly outperform models that combine multiple models at training time.

3 Our Approach

In this paper, we propose a method to apply stacking to statistical machine translation (SMT) and our method is the first to successfully exploit stacking for statistical machine translation. We use a standard statistical machine translation engine and produce multiple diverse models by partitioning the training set using the k-fold cross-validation technique. A diverse ensemble of weak systems is created by learning a model on each $k-1$ fold and tuning the statistical machine translation log-linear weights on the remaining fold. However, instead of learning a model on the output of base models as in (Wolpert, 1992), we combine hypotheses from the base models in the decoder with uniform weights. For the base learner, we use Kriya (Sankaran et al., 2012), an in-house hierarchical phrase-based machine translation system, to produce multiple weak models. These models are combined together using *Ensemble Decoding* (Razmara et al., 2012) to produce a strong model in the decoder. This method is briefly explained in next section.

3.1 Ensemble Decoding

SMT Log-linear models (Koehn, 2010) find the most likely target language output $\boldsymbol{e}$ given the source language input $\boldsymbol{f}$ using a vector of feature functions $\boldsymbol{\phi}$:

$$p(e|f) \propto \exp\Big(\mathbf{w} \cdot \boldsymbol{\phi}\Big)$$

[1] http://www.netflixprize.com/

Ensemble decoding combines several models dynamically at decoding time. The scores are combined for each partial hypothesis using a user-defined mixture operation $\otimes$ over component models.

$$p(e|f) \propto \exp\Big(\mathbf{w_1} \cdot \boldsymbol{\phi_1} \otimes \mathbf{w_2} \cdot \boldsymbol{\phi_2} \otimes \ldots\Big)$$

We previously successfully applied ensemble decoding to domain adaptation in SMT and showed that it performed better than approaches that pre-compute linear mixtures of different models (Razmara et al., 2012). Several mixture operations were proposed, allowing the user to encode belief about the relative strengths of the component models. These mixture operations receive two or more probabilities and return the mixture probability $p(\bar{e} \,|\, \bar{f})$ for each rule $\bar{e}, \bar{f}$ used in the decoder. Different options for these operations are:

- **Weighted Sum (wsum)** is defined as:

$$p(\bar{e} \,|\, \bar{f}) \;\;\propto\;\; \sum_{m}^{M} \lambda_m \exp\big(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\big)$$

  where m denotes the index of component models, M is the total number of them and λ_m is the weight for component m.

- **Weighted Max (wmax)** is defined as:

$$p(\bar{e} \,|\, \bar{f}) \;\;\propto\;\; \max_{m}\big(\lambda_m \exp\big(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\big)\big)$$

- **Prod** or **log-wsum** is defined as:

$$p(\bar{e} \,|\, \bar{f}) \;\;\propto\;\; \exp\Big(\sum_{m}^{M} \lambda_m \left(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\right)\Big)$$

- **Model Switching (Switch):** Each cell in the CKY chart is populated only by rules from one of the models and the other models' rules are discarded. Each component model is considered as an expert on different spans of the source. A binary indicator function $\delta(\bar{f}, m)$ picks a component model for each span:

$$\delta(\bar{f}, m) = \begin{cases} 1, & m = \underset{n \in M}{\text{argmax}} \;\; \psi(\bar{f}, n) \\ 0, & \text{otherwise} \end{cases}$$

  The criteria for choosing a model for each cell, $\psi(\bar{f}, n)$, could be based on max (SW:MAX), i.e. for each cell, the model that has the highest weighted score wins:

$$\psi(\bar{f}, n) = \lambda_n \max_{\bar{e}}(\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{\mathbf{e}}, \bar{\mathbf{f}}))$$

	Train size	Src tokens	Tgt tokens
Fr - En	0+dev	67K	58K
	10k+dev	365K	327K
	100k+dev	3M	2.8M
Es - En	0+dev	60K	58K
	10k+dev	341K	326K
	100k+dev	2.9M	2.8M

Table 1: Statistics of the training set for different systems and different language pairs.

Alternatively, we can pick the model with highest weighted sum of the probabilities of the rules (SW:SUM). This sum has to take into account the translation table limit (*ttl*), on the number of rules suggested by each model for each cell:

$$\psi(\bar{f}, n) = \lambda_n \sum_{\bar{e}} \exp\big(\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{\mathbf{e}}, \bar{\mathbf{f}})\big)$$

The probability of each phrase-pair $(\bar{e}, \bar{f})$ is then:

$$p(\bar{e} \,|\, \bar{f}) = \sum_{m}^{M} \delta(\bar{f}, m)\; p_m(\bar{e} \,|\, \bar{f})$$

4 Experiments & Results

We experimented with two language pairs: French to English and Spanish to English on the *Europarl* corpus (v7) (Koehn, 2005) and used ACL/WMT 2005 [2] data for dev and test sets.

For the base models, we used an in-house implementation of hierarchical phrase-based systems, Kriya (Sankaran et al., 2012), which uses the same features mentioned in (Chiang, 2005): forward and backward relative-frequency and lexical TM probabilities; LM; word, phrase and glue-rules penalty. GIZA++ (Och and Ney, 2003) has been used for word alignment with phrase length limit of 10. Feature weights were optimized using MERT (Och, 2003). We built a 5-gram language model on the English side of Europarl and used the Kneser-Ney smoothing method and SRILM (Stolcke, 2002) as the language model toolkit.

[2] http://www.statmt.org/wpt05/mt-shared-task/

Direction	k-fold	Resub	Mean	WSUM	WMAX	PROD	SW:MAX	SW:SUM
Fr - En	2	18.08	19.67	22.32	**22.48**	22.06	21.70	21.81
	4	18.08	21.80	23.14	23.48	**23.55**	22.83	22.95
	8	18.08	22.47	23.76	23.75	**23.78**	23.02	23.47
Es - En	2	18.61	19.23	**21.62**	21.33	21.49	21.48	21.51
	4	18.61	21.52	23.42	22.81	**22.91**	22.81	**22.92**
	8	18.61	22.20	23.69	**23.89**	23.51	22.92	23.26

Table 2: Testset BLEU scores when applying stacking on the devset only (using no specific training set).

Direction	Corpus	k-fold	Baseline	BMA	WSUM	WMAX	PROD	SW:MAX	SW:SUM
Fr - En	10k+dev	6	28.75	29.49	**29.87**	29.78	29.21	29.69	29.59
	100k+dev	11 / 51	29.53	29.75	34.00	**34.07**	33.11	34.05	33.96
Es - En	10k+dev	6	28.21	28.76	**29.59**	29.51	29.15	29.10	29.21
	100k+dev	11 / 51	33.25	33.44	**34.21**	34.00	33.17	34.19	**34.22**

Table 3: Testset BLEU scores when using 10k and 100k sentence training sets along with the devset.

4.1 Training on devset

We first consider the scenario in which there is no parallel data between a language pair except a small bi-text used as a devset. We use no specific training data and construct a SMT system completely on the devset by using our approach and compare to two different baselines. A natural baseline when having a limited parallel text is to do re-substitution validation where the model is trained on the whole devset and is tuned on the same set. This validation process suffers seriously from over-fitting. The second baseline is the mean of BLEU scores of all base models.

Table 2 summarizes the BLEU scores on the testset when using stacking only on the devset on two different language pairs. As the table shows, increasing the number of folds results in higher BLEU scores. However, doing such will generally lead to higher variance among base learners.

Figure 1 shows the BLEU score of each of the base models resulted from a 20-fold partitioning of the devset along with the strong models' BLEU scores. As the figure shows, the strong models are generally superior to the base models whose mean is represented as a horizontal line.

4.2 Training on train+dev

When we have some training data, we can use the cross-validation-style partitioning to create k splits. We then train a system on $k-1$ folds and tune on the devset. However, each system eventually wastes a fold of the training data. In order to take advantage of that remaining fold, we concatenate the devset to the training set and partition the whole union. In this way, we use all data available to us. We experimented with two sizes of training data: 10k sentence pairs and 100k, that with the addition of the devset, we have 12k and 102k sentence-pair corpora.

Table 1 summarizes statistics of the data sets used in this scenario. Table 3 reports the BLEU scores when using stacking on these two corpus sizes. The baselines are the conventional systems which are built on the training-set only and tuned on the devset as well as *Bayesian Model Averaging* (BMA, see §5). For the 100k+dev corpus, we sampled 11 partitions from all 51 possible partitions by taking every fifth partition as training data. The results in Table 3 show that stacking can improve over the baseline BLEU scores by up to 4 points.

Examining the performance of the different mixture operations, we can see that WSUM and WMAX typically outperform other mixture operations. Different mixture operations can be dominant in different language pairs and different sizes of training sets.

5 Related Work

Xiao et al. (2013) have applied both boosting and bagging on three different statistical machine translation engines: phrase-based (Koehn et al., 2003), hierarchical phrase-based (Chiang, 2005) and syntax-based (Galley et al., 2006) and showed SMT can benefit from these methods as well.

Duan et al. (2009) creates an ensemble of models by using feature subspace method in the machine learning literature (Ho, 1998). Each member of the ensemble is built by removing one non-LM feature in the log-linear framework or varying the order of language model. Finally they use a sentence-level system combination on the outputs of the base models to pick the best system for each

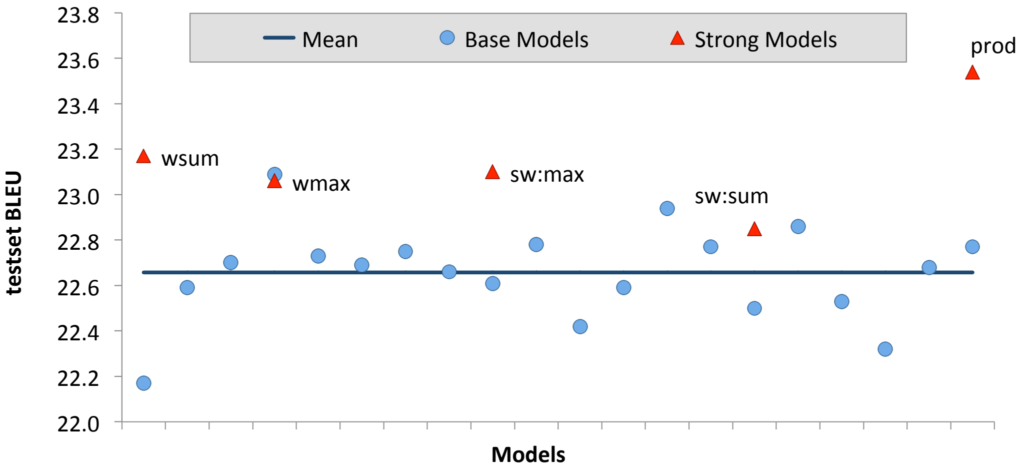

Figure 1: BLEU scores for all the base models and stacked models on the Fr-En devset with 20-fold cross validation. The horizontal line shows the mean of base models' scores.

sentence. Though, they do not combine the hypotheses search spaces of individual base models.

Our work is most similar to that of Duan et al. (2010) which uses *Bayesian model averaging* (BMA) (Hoeting et al., 1999) for SMT. They used sampling without replacement to create a number of base models whose phrase-tables are combined with that of the baseline (trained on the full training-set) using linear mixture models (Foster and Kuhn, 2007).

Our approach differs from this approach in a number of ways: *i)* we use cross-validation-style partitioning for creating training subsets while they do sampling without replacement (80% of the training set); *ii)* in our approach a number of base models are trained and tuned and they are combined on-the-fly in the decoder using *ensemble decoding* which has been shown to be more effective than offline combination of phrase-table-only features; *iii)* in Duan et al. (2010)'s method, each system gives up 20% of the training data in exchange for more diversity, but in contrast, our method not only uses all available data for training, but promotes diversity through allowing each model to tune on a different data set; *iv)* our approach takes advantage of held out data (the tuning set) in the training of base models which is beneficial especially when little parallel data is available or tuning/test sets and training sets are from different domains.

Empirical results (Table 3) also show that our approach outperforms the Bayesian model averaging approach (BMA).

6 Conclusion & Future Work

In this paper, we proposed a novel method on applying stacking to the statistical machine translation task. The results when using no, 10k and 100k sentence-pair training sets (along with a development set for tuning) show that stacking can yield an improvement of up to 4 BLEU points over conventionally trained SMT models which use a fixed training and tuning set.

Future work includes experimenting with larger training sets to investigate how useful this approach can be when having different sizes of training data.

References

Leo Breiman. 1996a. Bagging predictors. *Machine Learning*, 24(2):123–140, August.

Leo Breiman. 1996b. Stacked regressions. *Machine Learning*, 24(1):49–64, July.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Morristown, NJ, USA. ACL.

Nan Duan, Mu Li, Tong Xiao, and Ming Zhou. 2009. The feature subspace method for smt system combination. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1096–1104, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nan Duan, Hong Sun, and Ming Zhou. 2010. Translation model generalization using probability averaging for machine translation. In *Proceedings of the*

23rd International Conference on Computational Linguistics, COLING '10, pages 304–312, Stroudsburg, PA, USA. Association for Computational Linguistics.

André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, Hawaii, October. Association for Computational Linguistics.

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 128–135, Stroudsburg, PA, USA. ACL.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 961–968, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tin Kam Ho. 1998. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, August.

Jennifer A. Hoeting, David Madigan, Adrian E. Raftery, and Chris T. Volinsky. 1999. Bayesian Model Averaging: A Tutorial. *Statistical Science*, 14(4):382–401.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 127–133, Edmonton, May. NAACL.

P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.

Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition.

Antonio Lagarda and Francisco Casacuberta. 2008. Applying boosting to statistical machine translation. In *Annual Meeting of European Association for Machine Translation (EAMT)*, pages 88–96.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio, June. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.

Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *Proceedings of the 41th Annual Meeting of the ACL*, Sapporo, July. ACL.

Majid Razmara, George Foster, Baskaran Sankaran, and Anoop Sarkar. 2012. Mixing multiple translation models in statistical machine translation. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, pages 940–949. The Association for Computer Linguistics.

Erik F. Tjong Kim Sang. 2002. Memory-based shallow parsing. *J. Mach. Learn. Res.*, 2:559–594, March.

Baskaran Sankaran, Majid Razmara, and Anoop Sarkar. 2012. Kriya an end-to-end hierarchical phrase-based mt system. *The Prague Bulletin of Mathematical Linguistics*, 97(97), April.

Robert E. Schapire. 1990. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, July.

Linfeng Song, Haitao Mi, Yajuan Lü, and Qun Liu. 2011. Bagging-based system combination for domain adaption. In *Proceedings of the 13th Machine Translation Summit (MT Summit XIII)*, pages 293–299. International Association for Machine Translation, September.

Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.

Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble models for dependency parsing: cheap and good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 649–652, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nadi Tomeh, Alexandre Allauzen, Guillaume Wisniewski, and François Yvon. 2010. Refining word alignment with discriminative training. In *Proceedings of The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*.

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

Tong Xiao, Jingbo Zhu, Muhua Zhu, and Huizhen Wang. 2010. Boosting-based system combination for machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 739–748, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tong Xiao, Jingbo Zhu, and Tongran Liu. 2013. Bagging and boosting statistical machine translation systems. *Artificial Intelligence*, 195:496–527, February.