A Tale about PRO and Monsters

Preslav Nakov, Francisco Guzmán and Stephan Vogel
Qatar Computing Research Institute, Qatar Foundation
Tornado Tower, floor 10, PO box 5825
Doha, Qatar
{pnakov,fherrera,svogel}@qf.org.qa

Abstract

While experimenting with tuning on long sentences, we made an unexpected discovery: that PRO falls victim to *monsters* – overly long negative examples with very low BLEU+1 scores, which are unsuitable for learning and can cause testing BLEU to drop by several points absolute. We propose several effective ways to address the problem, using length- and BLEU+1-based cut-offs, outlier filters, stochastic sampling, and random acceptance. The best of these fixes not only slay and protect against monsters, but also yield higher stability for PRO as well as improved test-time BLEU scores. Thus, we recommend them to anybody using PRO, monster-believer or not.

1 Once Upon a Time...

For years, the standard way to do statistical machine translation parameter tuning has been to use minimum error-rate training, or MERT (Och, 2003). However, as researchers started using models with thousands of parameters, new scalable optimization algorithms such as MIRA (Watanabe et al., 2007; Chiang et al., 2008) and PRO (Hopkins and May, 2011) have emerged. As these algorithms are relatively new, they are still not quite well understood, and studying their properties is an active area of research.

For example, Nakov et al. (2012) have pointed out that PRO tends to generate translations that are consistently shorter than desired. They have blamed this on inadequate smoothing in PRO's optimization objective, namely sentence-level BLEU+1, and they have addressed the problem using more sensible smoothing. We wondered whether the issue could be partially relieved simply by tuning on longer sentences, for which the effect of smoothing would naturally be smaller.

To our surprise, tuning on the longer 50% of the tuning sentences had a disastrous effect on PRO, causing an absolute drop of three BLEU points on testing; at the same time, MERT and MIRA did not have such a problem. While investigating the reasons, we discovered hundreds of monsters creeping under PRO's surface...

Our tale continues as follows. We first explain what monsters are in Section 2, then we present a theory about how they can be slayed in Section 3, we put this theory to test in practice in Section 4, and we discuss some related efforts in Section 5. Finally, we present the moral of our tale, and we hint at some planned future battles in Section 6.

2 Monsters, Inc.

PRO uses pairwise ranking optimization, where the learning task is to classify pairs of hypotheses into correctly or incorrectly ordered (Hopkins and May, 2011). It searches for a vector of weights w such that higher evaluation metric scores correspond to higher model scores and vice versa. More formally, PRO looks for weights w such that $g(i,j) > g(i,j') \Leftrightarrow h_w(i,j) > h_w(i,j')$, where g is a local scoring function (typically, sentence-level BLEU+1) and h_w are the model scores for a given input sentence i and two candidate hypotheses j and j' that were obtained using w. If $g(i,j) > g(i,j')$, we will refer to j and j' as the positive and the negative example in the pair.

Learning good parameter values requires negative examples that are comparable to the positive ones. Instead, tuning on long sentences quickly introduces *monsters*, i.e., corrupted negative examples that are unsuitable for learning: they are (i) much longer than the respective positive examples and the references, and (ii) have very low BLEU+1 scores compared to the positive examples and in absolute terms. The low BLEU+1 means that PRO effectively has to learn from positive examples only.

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 12–17,
Sofia, Bulgaria, August 4-9 2013. ©2013 Association for Computational Linguistics

iter.	Avg. Lengths			Avg. BLEU+1	
	pos	neg	ref.	pos	neg
1	45.2	44.6	46.5	52.5	37.6
2	46.4	70.5	53.2	52.8	14.5
3	46.4	**261.0**	53.4	52.4	**2.19**
4	46.4	**250.0**	53.0	52.0	**2.30**
5	46.3	**248.0**	53.0	52.1	**2.34**
...	...	...	...	...	...
25	47.9	**229.0**	52.5	52.2	**2.81**

Table 1: PRO iterations, tuning on long sentences.

Table 1 shows an optimization run of PRO when tuning on long sentences. We can see monsters after iterations in which positive examples are on average longer than negative ones (e.g., iter. 1). As a result, PRO learns to generate longer sentences, but it overshoots too much (iter. 2), which gives rise to monsters. Ideally, the learning algorithm should be able to recover from overshooting. However, once monsters are encountered, they quickly start dominating, with no chance for PRO to recover since it accumulates n-best lists, and thus also monsters, over iterations. As a result, PRO keeps jumping up and down and converges to random values, as Figure 1 shows.

By default, PRO's parameters are averaged over iterations, and thus the final result is quite mediocre, but selecting the highest tuning score does not solve the problem either: for example, on Figure 1, PRO never achieves a BLEU better than that for the default initialization parameters.

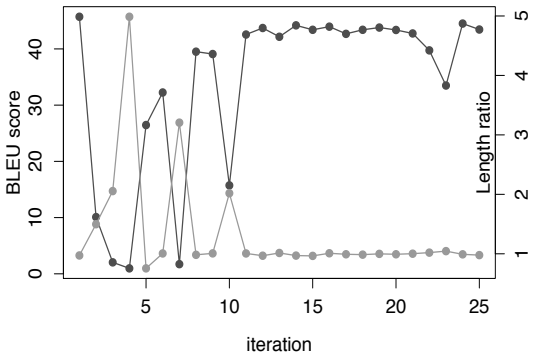

Figure 1: PRO tuning results on long sentences across iterations. The dark-gray line shows the tuning BLEU (left axis), the light-gray one is the hypothesis/reference length ratio (right axis).

Figure 2 shows the translations after iterations 1, 3 and 4; the last two are monsters. The monster at iteration 3 is potentially useful, but that at iteration 4 is clearly unsuitable as a negative example.

Optimizer	Objective	BLEU
PRO	sent-BLEU+1	**44.57**
MERT	corpus-BLEU	47.53
MIRA	pseudo-doc-BLEU	47.80
PRO ($\neq$ objective)	pseudo-doc-BLEU	**21.35**
MIRA ($\neq$ objective)	sent-BLEU+1	47.59
PRO, PC-smooth, ground	*fixed* sent-BLEU+1	**45.71**

Table 2: PRO vs. MERT vs. MIRA.

We also checked whether other popular optimizers yield very low BLEU scores at test time when tuned on long sentences. Lines 2-3 in Table 2 show that this is not the case for MERT and MIRA. Since they optimize objectives that are different from PRO's,[1] we further experimented with plugging MIRA's objective into PRO and PRO's objective into MIRA. The resulting MIRA scores were not much different from before, while PRO's score dropped even further; we also found monsters. Next, we applied the length fix for PRO proposed in (Nakov et al., 2012); this helped a bit, but still left PRO two BLEU points behind MERT[2] and MIRA, and the monsters did not go away. We can conclude that the monster problem is PRO-specific, cannot be blamed on the objective function, and is different from the length bias.

Note also that monsters are not specific to a dataset or language pair. We found them when tuning on the top-50% of WMT10 and testing on WMT11 for Spanish-English; this yielded a drop in BLEU from 29.63 (MERT) to 27.12 (PRO).

```
**REF**: but we have to close ranks with each other and realize that in
unity there is strength while in division there is weakness .
---------------------------------------------------------
**IT1**: but we are that we add our ranks to some of us and that we know
that in the strength and weakness in

**IT3**:, we are the but of the that that the , and , of ranks the the on
the the our the our the some of we can include , and , of to the of we know
the the our in of the of some people , force of the that that the in of the
that that the the weakness Union the the , and

**IT4**: namely Dr Heba Handossah and Dr Mona been pushed aside because a
larger story EU Ambassador to Egypt Ian Burg highlighted 've dragged us
backwards and dragged our speaking , never balme your defaulting a December
7th 1941 in Pearl Harbor ) we can include ranks will be joined by all 've
dragged us backwards and dragged our $ 3.8 billion in tourism income
proceeds Chamber are divided among themselves : some 've dragged us
backwards and dragged our were exaggerated . Al @-@ Hakim namely Dr Heba
Handossah and Dr Mona December 7th 1941 in Pearl Harbor ) cases might be
known to us December 7th 1941 in Pearl Harbor ) platform depends on
combating all liberal policies Track and Field Federation shortened strength
as well face several challenges , namely Dr Heba Handossah and Dr Mona
platform depends on combating all liberal policies the report forecast that
the weak structure
```

Figure 2: Example reference translation and hypothesis translations after iterations 1, 3 and 4. The last two hypotheses are monsters.

[1] See (Cherry and Foster, 2012) for details on objectives.

[2] Also, using PRO to initialize MERT, as implemented in Moses, yields 46.52 BLEU and monsters, but using MERT to initialize PRO yields 47.55 and no monsters.

3 Slaying Monsters: Theory

Below we explain what monsters are and where they come from. Then, we propose various monster slaying techniques to be applied during PRO's selection and acceptance steps.

3.1 What is PRO?

PRO is a batch optimizer that iterates between (i) *translation*: using the current parameter values, generate k-best translations, and (ii) *optimization*: using the translations from all previous iterations, find new parameter values. The optimization step has four substeps:

1. **Sampling:** For each sentence, sample uniformly at random $\Gamma = 5000$ pairs from the set of all candidate translations for that sentence from all previous iterations.

2. **Selection:** From these sampled pairs, select those for which the absolute difference between their BLEU+1 scores is higher than $\alpha = 0.05$ (note: this is 5 BLEU+1 points).

3. **Acceptance:** For each sentence, accept the $\Xi = 50$ selected pairs with the highest absolute difference in their BLEU+1 scores.

4. **Learning:** Assemble the accepted pairs for all sentences into a single set and use it to train a ranker to prefer the higher-scoring sentence in each pair.

We believe that monsters are nurtured by PRO's selection and acceptance policies. PRO's selection step filters pairs involving hypotheses that differ by less than five BLEU+1 points, but it does not cut-off ones that differ too much based on BLEU+1 or length. PRO's acceptance step selects $\Xi = 50$ pairs with the highest BLEU+1 differentials, which creates breeding ground for monsters since these pairs are very likely to include one monster and one good hypothesis.

Below we discuss monster slaying geared towards the selection and acceptance steps of PRO.

3.2 Slaying at Selection

In the selection step, PRO filters pairs for which the difference in BLEU+1 is *less* than five points, but it has no cut-off on the *maximum* BLEU+1 differentials nor cut-offs based on absolute length or difference in *length*. Here, we propose several selection filters, both deterministic and probabilistic.

Cut-offs. A cut-off is a deterministic rule that filters out pairs that do not comply with some criteria. We experiment with a maximal cut-off on (a) the difference in BLEU+1 scores and (b) the difference in lengths. These are relative cut-offs because they refer to the pair, but absolute cut-offs that apply to each of the elements in the pair are also possible (not explored here). Cut-offs (a) and (b) slay monsters by not allowing the negative examples to get much worse in BLEU+1 or in length than the positive example in the pair.

Filtering outliers. Outliers are rare or extreme observations in a sample. We assume normal distribution of the BLEU+1 scores (or of the lengths) of the translation hypotheses for the same source sentence, and we define as outliers hypotheses whose BLEU+1 (or length) is more than λ standard deviations away from the sample average. We apply the outlier filter to both the positive and the negative example in a pair, but it is more important for the latter. We experiment with values of λ like 2 and 3. This filtering slays monsters because they are likely outliers. However, it will not work if the population gets riddled with monsters, in which case they would become the norm.

Stochastic sampling. Instead of filtering extreme examples, we can randomly sample pairs according to their probability of being typical. Let us assume that the values of the local scoring functions, i.e., the BLEU+1 scores, are distributed normally: $g(i,j) \sim N(\mu, \sigma^2)$. Given a sample of hypothesis translations $\{j\}$ of the same source sentence i, we can estimate σ empirically. Then, the difference $\Delta = g(i,j) - g(i,j')$ would be distributed normally with mean zero and variance $2\sigma^2$. Now, given a pair of examples, we can calculate their Δ, and we can choose to select the pair with some probability, according to $N(0, 2\sigma^2)$.

3.3 Slaying at Acceptance

Another problem is caused by the acceptance mechanism of PRO: among all selected pairs, it accepts the top-Ξ with the highest BLEU+1 differentials. It is easy to see that these differentials are highest for nonmonster–monster pairs if such pairs exist. One way to avoid focusing primarily on such pairs is to accept a random set of Ξ pairs, among the ones that survived the selection step. One possible caveat is that we can lose some of the discriminative power of PRO by focusing on examples that are not different enough.

		TESTING		TUNING (run 1, it. 25, avg.)					TEST(tune:full)	
		Avg. for 3 reruns		Lengths			BLEU+1		Avg. for 3 reruns	
	PRO fix	**BLEU**	**StdDev**	**Pos**	**Neg**	**Ref**	**Pos**	**Neg**	**BLEU**	**StdDev**
	PRO (baseline)	44.70	0.266	47.9	**229.0**	52.5	52.2	**2.8**	47.80	0.052
Max diff. cut-off	BLEU+1 max=10 †	**47.94**	**0.165**	47.9	49.6	49.4	49.4	39.9	47.77	**0.035**
	BLEU+1 max=20 †	47.73	**0.136**	47.7	55.5	51.1	49.8	32.7	**47.85**	**0.049**
	LEN max=5 †	**48.09**	**0.021**	46.8	47.0	47.9	52.9	37.8	47.73	**0.051**
	LEN max=10 †	**47.99**	**0.025**	47.3	48.5	48.7	52.5	35.6	**47.80**	0.056
Outliers	BLEU+1 λ=2.0 †	**48.05**	**0.119**	46.8	47.2	47.7	52.2	39.5	47.47	0.090
	BLEU+1 λ=3.0	47.12	1.348	47.6	**168.0**	53.0	51.7	**3.9**	47.53	**0.038**
	LEN λ=2.0	46.68	2.005	49.3	**82.7**	53.1	52.3	**5.3**	47.49	0.085
	LEN λ=3.0	47.02	0.727	48.2	**163.0**	51.4	51.4	**4.2**	47.65	0.096
Stoch. sampl.	Δ BLEU+1	46.33	1.000	46.8	**216.0**	53.3	53.1	**2.4**	47.74	**0.035**
	Δ LEN	46.36	1.281	47.4	**201.0**	52.9	53.4	**2.9**	47.78	0.081

Table 3: Some fixes to PRO (select pairs with highest BLEU+1 differential, also require at least 5 BLEU+1 points difference). A dagger (†) indicates selection fixes that successfully get rid of monsters.

4 Attacking Monsters: Practice

Below, we first present our general experimental setup. Then, we present the results for the various selection alternatives, both with the original acceptance strategy and with random acceptance.

4.1 Experimental Setup

We used a phrase-based SMT model (Koehn et al., 2003) as implemented in the Moses toolkit (Koehn et al., 2007). We trained on all Arabic-English data for NIST 2012 except for UN, we tuned on (the longest-50% of) the MT06 sentences, and we tested on MT09. We used the MADA ATB segmentation for Arabic (Roth et al., 2008) and truecasing for English, phrases of maximal length 7, Kneser-Ney smoothing, and lexicalized reordering (Koehn et al., 2005), and a 5-gram language model, trained on GigaWord v.5 using KenLM (Heafield, 2011). We dropped unknown words both at tuning and testing, and we used minimum Bayes risk decoding at testing (Kumar and Byrne, 2004). We evaluated the output with NIST's scoring tool v.13a, cased.

We used the Moses implementations of MERT, PRO and batch MIRA, with the –return-best-dev parameter for the latter. We ran these optimizers for up to 25 iterations and we used 1000-best lists.

For stability (Foster and Kuhn, 2009), we performed three reruns of each experiment (tuning + evaluation), and we report averaged scores.

4.2 Selection Alternatives

Table 3 presents the results for different selection alternatives. The first two columns show the testing results: average BLEU and standard deviation over three reruns.

The following five columns show statistics about the last iteration (it. 25) of PRO's tuning for the worst rerun: average lengths of the positive and the negative examples and average effective reference length, followed by average BLEU+1 scores for the positive and the negative examples in the pairs. The last two columns present the results when tuning on the full tuning set. These are included to verify the behavior of PRO in a non-monster prone environment.

We can see in Table 3 that all selection mechanisms considerably improve BLEU compared to the baseline PRO, by 2-3 BLEU points. However, not every selection alternative gets rid of monsters, which can be seen by the large lengths and low BLEU+1 for the negative examples (in bold).

The max cut-offs for BLEU+1 and for lengths both slay the monsters, but the latter yields much lower standard deviation (thirteen times lower than for the baseline PRO!), thus considerably increasing PRO's stability. On the full dataset, BLEU scores are about the same as for the original PRO (with small improvement for BLEU+1 max=20), but the standard deviations are slightly better.

Rejecting outliers using BLEU+1 and $\lambda = 3$ is not strong enough to filter out monsters, but making this criterion more strict by setting $\lambda = 2$, yields competitive BLEU and kills the monsters.

Rejecting outliers based on length does not work as effectively though. We can think of two possible reasons: (i) lengths are not normally distributed, they are more Poisson-like, and (ii) the acceptance criterion is based on the top-Ξ differentials based on BLEU+1, not based on length.

On the full dataset, rejecting outliers, BLEU+1 and length, yields lower BLEU and less stability.

		TESTING		TUNING (run 1, it. 25, avg.)					TEST(tune:full)	
		Avg. for 3 reruns		Lengths			BLEU+1		Avg. for 3 reruns	
	PRO fix	**BLEU**	**StdDev**	**Pos**	**Neg**	**Ref**	**Pos**	**Neg**	**BLEU**	**StdDev**
	PRO (baseline)	44.70	0.266	47.9	**229.0**	52.5	52.2	**2.8**	47.80	0.052
Rand. accept	PRO, rand ††	47.87	0.147	47.7	48.5	48.70	47.7	42.9	47.59	0.114
Outliers	BLEU+1 λ=2.0, rand*	47.85	**0.078**	48.2	48.4	48.9	47.5	43.6	**47.62**	**0.091**
	BLEU+1 λ=3.0, rand	**47.97**	**0.168**	47.6	47.6	48.4	47.8	43.6	47.44	**0.070**
	LEN λ=2.0, rand*	47.69	**0.114**	47.8	47.8	48.6	47.9	43.6	47.48	**0.046**
	LEN λ=3.0, rand	47.89	0.235	47.8	48.0	48.7	47.7	43.1	**47.64**	**0.090**
Stoch. sampl.	Δ BLEU+1, rand*	**47.99**	**0.087**	47.9	48.0	48.7	47.8	43.5	**47.67**	**0.096**
	Δ LEN, rand*	**47.94**	**0.060**	47.8	47.9	48.6	47.8	43.6	**47.65**	**0.097**

Table 4: More fixes to PRO (with random acceptance, no minimum BLEU+1). The (††) indicates that random acceptance kills monsters. The asterisk (*) indicates improved stability over random acceptance.

Reasons (i) and (ii) arguably also apply to stochastic sampling of differentials (for BLEU+1 or for length), which fails to kill the monsters, maybe because it gives them some probability of being selected by design. To alleviate this, we test the above settings with random acceptance.

4.3 Random Acceptance

Table 4 shows the results for accepting training pairs for PRO uniformly at random. To eliminate possible biases, we also removed the min=0.05 BLEU+1 selection criterion. Surprisingly, this setup effectively eliminated the monster problem. Further coupling this with the distributional criteria can also yield increased stability, and even small further increase in test BLEU. For instance, rejecting BLEU outliers with $\lambda = 2$ yields comparable average test BLEU, but with only half the standard deviation.

On the other hand, using the stochastic sampling of differentials based on either BLEU+1 or lengths improves the test BLEU score while increasing the stability across runs. The random acceptance has a caveat though: it generally decreases the discriminative power of PRO, yielding worse results when tuning on the full, nonmonster prone tuning dataset. Stochastic selection does help to alleviate this problem. Yet, the results are not as good as when using a max cut-off for the length. Therefore, we recommend using the latter as a default setting.

5 Related Work

We are not aware of previous work that discusses the issue of monsters, but there has been work on a different, length problem with PRO (Nakov et al., 2012). We have seen that its solution, fix the smoothing in BLEU+1, did not work for us.

The stability of MERT has been improved using regularization (Cer et al., 2008), random restarts (Moore and Quirk, 2008), multiple replications (Clark et al., 2011), and parameter aggregation (Cettolo et al., 2011).

With the emergence of new optimization techniques, there have been studies that compare stability between MIRA–MERT (Chiang et al., 2008; Chiang et al., 2009; Cherry and Foster, 2012), PRO–MERT (Hopkins and May, 2011), MIRA–PRO–MERT (Cherry and Foster, 2012; Gimpel and Smith, 2012; Nakov et al., 2012).

Pathological verbosity can be an issue when tuning MERT on recall-oriented metrics such as METEOR (Lavie and Denkowski, 2009; Denkowski and Lavie, 2011). Large variance between the results obtained with MIRA has also been reported (Simianer et al., 2012). However, none of this work has focused on monsters.

6 Tale's Moral and Future Battles

We have studied a problem with PRO, namely that it can fall victim to monsters, overly long negative examples with very low BLEU+1 scores, which are unsuitable for learning. We have proposed several effective ways to address this problem, based on length- and BLEU+1-based cut-offs, outlier filters and stochastic sampling. The best of these fixes have not only slayed the monsters, but have also brought much higher stability to PRO as well as improved test-time BLEU scores. These benefits are less visible on the full dataset, but we still recommend them to everybody who uses PRO as protection against monsters. Monsters are inherent in PRO; they just do not always take over.

In future work, we plan a deeper look at the mechanism of monster creation in PRO and its possible connection to PRO's length bias.

References

Daniel Cer, Daniel Jurafsky, and Christopher Manning. 2008. Regularization and search for minimum error rate training. In *Proc. of Workshop on Statistical Machine Translation*, WMT '08, pages 26–34.

Mauro Cettolo, Nicola Bertoldi, and Marcello Federico. 2011. Methods for smoothing the optimizer instability in SMT. *MT Summit XIII: the Machine Translation Summit*, pages 32–39.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT '12, pages 427–436.

David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 224–233.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT '09, pages 218–226.

Jonathan Clark, Chris Dyer, Alon Lavie, and Noah Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the Meeting of the Association for Computational Linguistics*, ACL '11, pages 176–181.

Michael Denkowski and Alon Lavie. 2011. Meteor-tuned phrase-based SMT: CMU French-English and Haitian-English systems for WMT 2011. Technical report, CMU-LTI-11-011, Language Technologies Institute, Carnegie Mellon University.

George Foster and Roland Kuhn. 2009. Stabilizing minimum error rate training. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '09, pages 242–249.

Kevin Gimpel and Noah Smith. 2012. Structured ramp loss minimization for machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT '12, pages 221–231.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Workshop on Statistical Machine Translation*, WMT '11, pages 187–197.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1352–1362.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, HLT-NAACL '03, pages 48–54.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT '05.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of the Meeting of the Association for Computational Linguistics*, ACL '07, pages 177–180.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, HLT-NAACL '04, pages 169–176.

Alon Lavie and Michael Denkowski. 2009. The METEOR metric for automatic evaluation of machine translation. *Machine Translation*, 23:105–115.

Robert Moore and Chris Quirk. 2008. Random restarts in minimum error rate training for statistical machine translation. In *Proceedings of the International Conference on Computational Linguistics*, COLING '08, pages 585–592.

Preslav Nakov, Francisco Guzmán, and Stephan Vogel. 2012. Optimizing for sentence-level BLEU+1 yields short translations. In *Proceedings of the International Conference on Computational Linguistics*, COLING '12, pages 1979–1994.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Meeting of the Association for Computational Linguistics*, ACL '03, pages 160–167.

Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of the Meeting of the Association for Computational Linguistics*, ACL '08, pages 117–120.

Patrick Simianer, Stefan Riezler, and Chris Dyer. 2012. Joint feature selection in distributed stochastic learning for large-scale discriminative training in smt. In *Proceedings of the Meeting of the Association for Computational Linguistics*, ACL '12, pages 11–21.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '07, pages 764–773.