Learning to Translate with Multiple Objectives

Kevin Duh* **Katsuhito Sudoh** **Xianchao Wu** **Hajime Tsukada** **Masaaki Nagata**
NTT Communication Science Laboratories
2-4 Hikari-dai, Seika-cho, Kyoto 619-0237, JAPAN
kevinduh@is.naist.jp, lastname.firstname@lab.ntt.co.jp

Abstract

We introduce an approach to optimize a machine translation (MT) system on multiple metrics *simultaneously*. Different metrics (e.g. BLEU, TER) focus on different aspects of translation quality; our multi-objective approach leverages these diverse aspects to improve overall quality.

Our approach is based on the theory of Pareto Optimality. It is simple to implement on top of existing single-objective optimization methods (e.g. MERT, PRO) and outperforms ad hoc alternatives based on linear-combination of metrics. We also discuss the issue of metric tunability and show that our Pareto approach is more effective in incorporating new metrics from MT evaluation for MT optimization.

1 Introduction

Weight optimization is an important step in building machine translation (MT) systems. Discriminative optimization methods such as MERT (Och, 2003), MIRA (Crammer et al., 2006), PRO (Hopkins and May, 2011), and Downhill-Simplex (Nelder and Mead, 1965) have been influential in improving MT systems in recent years. These methods are effective because they tune the system to maximize an automatic evaluation metric such as BLEU, which serve as surrogate objective for translation quality.

However, we know that a single metric such as BLEU is not enough. *Ideally*, we want to tune towards an automatic metric that has *perfect* correlation with human judgments of translation quality. While many alternatives have been proposed, such a perfect evaluation metric remains elusive.

As a result, many MT evaluation campaigns now report multiple evaluation metrics (Callison-Burch et al., 2011; Paul, 2010). Different evaluation metrics focus on different aspects of translation quality. For example, while BLEU (Papineni et al., 2002) focuses on word-based n-gram precision, METEOR (Lavie and Agarwal, 2007) allows for stem/synonym matching and incorporates recall. TER (Snover et al., 2006) allows arbitrary chunk movements, while permutation metrics like RIBES (Isozaki et al., 2010; Birch et al., 2010) measure deviation in word order. Syntax (Owczarzak et al., 2007) and semantics (Pado et al., 2009) also help. Arguably, all these metrics correspond to our intuitions on what is a good translation.

The current approach of optimizing MT towards a single metric runs the risk of sacrificing other metrics. Can we really claim that a system is good if it has high BLEU, but very low METEOR? Similarly, is a high-METEOR low-BLEU system desirable? Our goal is to propose a multi-objective optimization method that avoids "overfitting to a single metric". We want to build a MT system that does well with respect to many aspects of translation quality.

In general, we cannot expect to improve multiple metrics jointly if there are some inherent tradeoffs. We therefore need to define the notion of Pareto Optimality (Pareto, 1906), which characterizes this tradeoff in a rigorous way and distinguishes the set of equally good solutions. We will describe Pareto Optimality in detail later, but roughly speaking, a

*Now at Nara Institute of Science & Technology (NAIST)

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 1–10,
Jeju, Republic of Korea, 8-14 July 2012.

hypothesis is pareto-optimal if there exist no other hypothesis better in all metrics. The contribution of this paper is two-fold:

- We introduce **PMO** (*P*areto-based *M*ulti-objective *O*ptimization), a general approach for learning with multiple metrics. Existing single-objective methods can be easily extended to multi-objective using PMO.
- We show that PMO outperforms the alternative (single-objective optimization of linearly-combined metrics) in multi-objective space, and especially obtains stronger results for metrics that may be difficult to tune individually.

In the following, we first explain the theory of Pareto Optimality (Section 2), and then use it to build up our proposed PMO approach (Section 3). Experiments on NIST Chinese-English and PubMed English-Japanese translation using BLEU, TER, and RIBES are presented in Section 4. We conclude by discussing related work (Section 5) and opportunities/limitations (Section 6).

2 Theory of Pareto Optimality

2.1 Definitions and Concepts

The idea of Pareto optimality comes originally from economics (Pareto, 1906), where the goal is to characterize situations when a change in allocation of goods does not make anybody worse off. Here, we will explain it in terms of MT:

Let $h \in L$ be a hypothesis from an N-best list L. We have a total of K different metrics $M_k(h)$ for evaluating the quality of h. Without loss of generality, we assume metric scores are bounded between 0 and 1, with 1 being perfect. Each hypothesis h can be mapped to a K-dimensional vector $M(h) = [M_1(h); M_2(h); ...; M_K(h)]$. For example, suppose $K = 2$, $M_1(h)$ computes the BLEU score, and $M_2(h)$ gives the METEOR score of h. Figure 1 illustrates the set of vectors $\{M(h)\}$ in a 10-best list.

For two hypotheses h_1, h_2, we write $M(h_1) > M(h_2)$ if h_1 is *better than* h_2 in all metrics, and $M(h_1) \geq M(h_2)$ if h_1 is *better than or equal to* h_2 in all metrics. When $M(h_1) \geq M(h_2)$ and $M_k(h_1) > M_k(h_2)$ for at least one metric k, we say that h_1 *dominates* h_2 and write $M(h_1) \triangleright M(h_2)$.

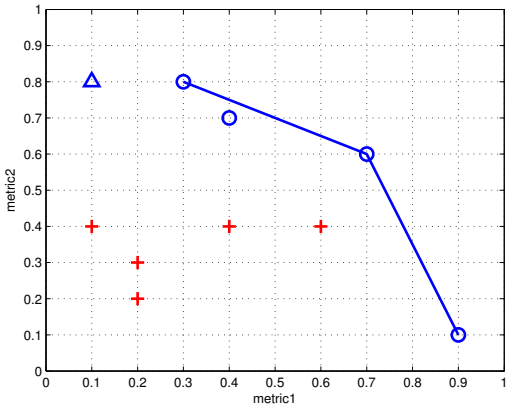

Figure 1: Illustration of Pareto Frontier. Ten hypotheses are plotted by their scores in two metrics. Hypotheses indicated by a circle (o) are pareto-optimal, while those indicated by a plus (+) are not. The line shows the convex hull, which attains only a subset of pareto-optimal points. The triangle (△) is a point that is weakly pareto-optimal but not pareto-optimal.

Definition 1. *Pareto Optimal: A hypothesis $h^* \in L$ is pareto-optimal iff there does not exist another hypothesis $h \in L$ such that $M(h) \triangleright M(h^*)$.*

In Figure 1, the hypotheses indicated by circle (o) are pareto-optimal, while those with plus (+) are not. To visualize this, take for instance the pareto-optimal point (0.4,0.7). There is no other point with either (metric1 > 0.4 and metric2 ≥ 0.7), or (metric1 ≥ 0.4 and metric2 > 0.7). On the other hand, the non-pareto point (0.6,0.4) is "dominated" by another point (0.7,0.6), because for metric1: $0.7 > 0.6$ and for metric2: $0.6 > 0.4$.

There is another definition of optimality, which disregards ties and may be easier to visualize:

Definition 2. *Weakly Pareto Optimal: A hypothesis $h^* \in L$ is weakly pareto-optimal iff there is no other hypothesis $h \in L$ such that $M(h) > M(h^*)$.*

Weakly pareto-optimal points are a superset of pareto-optimal points. A hypothesis is weakly pareto-optimal if there is no other hypothesis that improves all the metrics; a hypothesis is pareto-optimal if there is no other hypothesis that improves at least one metric without detriment to other metrics. In Figure 1, point (0.1,0.8) is weakly pareto-optimal but not pareto-optimal, because of the competing point (0.3,0.8). Here we focus on pareto-optimality, but note our algorithms can be easily

modified for weakly pareto-optimality. Finally, we can introduce the key concept used in our proposed PMO approach:

Definition 3. *Pareto Frontier: Given an N-best list L, the set of all pareto-optimal hypotheses $h \in L$ is called the Pareto Frontier.*

The Pareto Frontier has two desirable properties from the multi-objective optimization perspective:

1. Hypotheses on the Frontier are equivalently good in the Pareto sense.
2. For each hypothesis not on the Frontier, there is always a better (pareto-optimal) hypothesis.

This provides a principled approach to optimization: i.e. optimizing towards points on the Frontier and away from those that are not, and giving no preference to different pareto-optimal hypotheses.

2.2 Reduction to Linear Combination

Multi-objective problems can be formulated as:

$$\arg\max_{w} \; [M_1(h); M_2(h); \ldots; M_k(h)] \quad (1)$$
$$\text{where } h = \texttt{Decode}(w, f)$$

Here, the MT system's `Decode` function, parameterized by weight vector w, takes in a foreign sentence f and returns a translated hypothesis h. The `argmax` operates in vector space and our goal is to find w leading to hypotheses on the Pareto Frontier.

In the study of Pareto Optimality, one central question is: To what extent can multi-objective problems be solved by single-objective methods? Equation 1 can be *reduced* to a single-objective problem by scalarizing the vector $[M_1(h); \ldots; M_k(h)]$ with a linear combination:

$$\arg\max_{w} \; \sum_{k=1}^{K} p_k M_k(h) \quad (2)$$
$$\text{where } h = \texttt{Decode}(w, f)$$

Here, p_k are positive real numbers indicating the relative importance of each metric (without loss of generality, assume $\sum_k p_k = 1$). Are the solutions to Eq. 2 also solutions to Eq. 1 (i.e. pareto-optimal) and vice-versa? The theory says:

Theorem 1. *Sufficient Condition: If w^* is solution to Eq. 2, then it is weakly pareto-optimal. Further, if w^* is unique, then it is pareto-optimal.*

Theorem 2. *No Necessary Condition: There may exist solutions to Eq. 1 that cannot be achieved by Eq. 2, irregardless of any setting of $\{p_k\}$.*

Theorem 1 is a positive result asserting that linear combination can give pareto-optimal solutions. However, Theorem 2 states the limits: in particular, Eq. 2 attains only pareto-optimal points that are on the convex hull. This is illustrated in Figure 1: imagine sweeping all values of $p_1 = [0, 1]$ and $p_2 = 1 - p_1$ and recording the set of hypotheses that maximizes $\sum_k p_k M_k(h)$. For $0.6 < p_1 \leq 1$ we get $h = (0.9, 0.1)$, for $p_1 = 0.6$ we get $(0.7, 0.6)$, and for $0 < p_1 < 0.6$ we get $(0.4, 0.8)$. At no setting of p_1 do we attain $h = (0.4, 0.7)$ which is also pareto-optimal but not on the convex hull.[1] This may have ramifications for issues like metric tunability and local optima. To summarize, linear-combination is reasonable but has limitations. Our proposed approach will instead *directly* solve Eq. 1.

Pareto Optimality and multi-objective optimization is a deep field with active inquiry in engineering, operations research, economics, etc. For the interested reader, we recommend the survey by Marler and Arora (2004) and books by (Sawaragi et al., 1985; Miettinen, 1998).

3 Multi-objective Algorithms

3.1 Computing the Pareto Frontier

Our PMO approach will need to compute the Pareto Frontier for potentially large sets of points, so we first describe how this can be done efficiently. Given a set of N vectors $\{M(h)\}$ from an N-best list L, our goal is extract the subset that are pareto-optimal.

Here we present an algorithm based on *iterative filtering*, in our opinion the simplest algorithm to understand and implement. The strategy is to loop through the list L, keeping track of any dominant points. Given a dominant point, it is easy to filter out many points that are dominated by it. After successive rounds, any remaining points that are not fil-

[1] We note that scalarization by *exponentiated*-combination $\sum_k p_k M_k(h)^q$, for a suitable $q > 0$, does satisfy necessary conditions for pareto optimality. However the proper tuning of q is not known a priori. See (Miettinen, 1998) for theorem proofs.

Algorithm 1 FindParetoFrontier

Input: $\{M(h)\}, h \in L$
Output: All pareto-optimal points of $\{M(h)\}$
1: $\mathcal{F} = \emptyset$
2: **while** L is not empty **do**
3: h^* = shift(L)
4: **for** each h in L **do**
5: if $(M(h^*) \triangleright M(h))$: remove h from L
6: else if $(M(h) \triangleright M(h^*))$: remove h from L; set $h^* = h$
7: **end for**
8: Add h^* to Frontier Set $\mathcal{F}$
9: **for** each h in L **do**
10: if $(M(h^*) \triangleright M(h))$: remove h from L
11: **end for**
12: **end while**
13: Return $\mathcal{F}$

tered are necessarily pareto-optimal. Algorithm 1 shows the pseudocode. In line 3, we take a point h^* and check if it is dominating or dominated in the for-loop (lines 4-8). At least one pareto-optimal point will be found by line 8. The second loop (lines 9-11) further filters the list for points that are dominated by h^* but iterated before h^* in the first for-loop.

The outer while-loop stops exactly after P iterations, where P is the actual number of pareto-optimal points in L. Each inner loop costs $O(KN)$ so the total complexity is $O(PKN)$. Since $P \leq N$ with the actual value depending on the probability distribution of $\{M(h)\}$, the worst-case run-time is $O(KN^2)$. For a survey of various Pareto algorithms, refer to (Godfrey et al., 2007). The algorithm we described here is borrowed from the database literature in what is known as skyline operators.[2]

3.2 PMO-PRO Algorithm

We are now ready to present an algorithm for multi-objective optimization. As we will see, it can be seen as a generalization of the pairwise ranking optimization (PRO) of (Hopkins and May, 2011), so we call it PMO-PRO. PMO-PRO approach works by iteratively decoding-and-optimizing on the devset, similar to many MT optimization methods. The main difference is that rather than trying to maximize a single metric, we maximize the number of pareto points, in order to expand the Pareto Frontier

We will explain PMO-PRO in terms of the pseudo-code shown in Algorithm 2. For each sentence pair (f, e) in the devset, we first generate an N-best list $L \equiv \{h\}$ using the current weight vector w (line 5). In line 6, we evaluate each hypothesis h with respect to the K metrics, giving a set of K-dimensional vectors $\{M(h)\}$.

Lines 7-8 is the critical part: it gives a "label" to each hypothesis, based on whether it is in the Pareto Frontier. In particular, first we call `FindParetoFrontier` (Algorithm 1), which returns a set of pareto hypotheses; pareto-optimal hypotheses will get label 1 while non-optimal hypotheses will get label 0. This information is added to the training set $\mathcal{T}$ (line 8), which is then optimized by any conventional subroutine in line 10. We will follow PRO in using a pairwise classifier in line 10, which finds w^* that separates hypotheses with labels 1 vs. 0. In essence, this is the trick we employ to directly optimize on the Pareto Frontier. If we had used BLEU scores rather than the $\{0, 1\}$ labels in line 8, the entire PMO-PRO algorithm would revert to single-objective PRO.

By definition, there is no single "best" result for multi-objective optimization, so we collect all weights and return the Pareto-optimal set. In line 13 we evaluate each weight w on K metrics across the entire corpus and call `FindParetoFrontier` in line 14.[3] This choice highlights an interesting change of philosophy: While setting $\{p_k\}$ in linear-combination forces the designer to make an *a priori* preference among metrics prior to optimization, the PMO strategy is to optimize first agnostically and *a posteriori* let the designer choose among a set of weights. Arguably it is easier to choose among solutions based on their evaluation scores rather than devising exact values for $\{p_k\}$.

3.3 Discussion

Variants: In practice we find that a slight modification of line 8 in Algorithm 2 leads to more sta-

[2]The inquisitive reader may wonder how is Pareto related to databases. The motivation is to incorporate preferences into relational queries(Börzsönyi et al., 2001). For $K = 2$ metrics, they also present an alternative faster O(N logN) algorithm by first topologically sorting along the 2 dimensions. All dominated points can be filtered by one-pass by comparing with the most-recent dominating point.

[3]Note this is the same FindParetoFrontier algorithm as used in line 7. Both operate on sets of *points* in K-dimensional space, induced from either weights $\{w\}$ or hypotheses $\{h\}$.

Algorithm 2 Proposed PMO-PRO algorithm

Input: Devset, max number of iterations I
Output: A set of (pareto-optimal) weight vectors
1: Initialize w. Let $\mathcal{W} = \emptyset$.
2: **for** $i = 1$ to I **do**
3: Let $\mathcal{T} = \emptyset$.
4: **for** each (f, e) in devset **do**
5: $\{h\}$ =DecodeNbest(w,f)
6: $\{M(h)\}$=EvalMetricsOnSentence($\{h\}, e$)
7: $\{f\}$ =FindParetoFrontier($\{M(h)\}$)
8: foreach $h \in \{h\}$:
if $h \in \{f\}$, set l=1, else l=0; Add (l, h) to $\mathcal{T}$
9: **end for**
10: w^*=OptimizationSubroutine($\mathcal{T}, w$)
11: Add w^* to $\mathcal{W}$; Set $w = w^*$.
12: **end for**
13: $M(w)$ =EvalMetricsOnCorpus(w,devset) $\forall w \in \mathcal{W}$
14: Return FindParetoFrontier($\{M(w)\}$)

ble results for PMO-PRO: for non-pareto hypotheses $h \notin \{f\}$, we set label $l = \sum_k M_k(h)/K$ instead of $l = 0$, so the method not only learns to discriminate pareto vs. non-pareto but also also learns to discriminate among competing non-pareto points. Also, like other MT works, in line 5 the N-best list is concatenated to N-best lists from previous iterations, so $\{h\}$ is a set with $i \cdot N$ elements.

General PMO Approach: The strategy we outlined in Section 3.2 can be easily applied to other MT optimization techniques. For example, by replacing the optimization subroutine (line 10, Algorithm 2) with a Powell search (Och, 2003), one can get PMO-MERT[4]. Alternatively, by using the large-margin optimizer in (Chiang et al., 2009) and moving it into the for-each loop (lines 4-9), one can get an online algorithm such PMO-MIRA. Virtually all MT optimization algorithms have a place where metric scores feedback into the optimization procedure; the idea of PMO is to replace these raw scores with labels derived from Pareto optimality.

4 Experiments

4.1 Evaluation Methodology

We experiment with two datasets: (1) The **PubMed** task is English-to-Japanese translation of scientific abstracts. As metrics we use BLEU and RIBES (which demonstrated good human correlation in this language pair (Goto et al., 2011)). (2) The **NIST** task is Chinese-to-English translation with OpenMT08 training data and MT06 as devset. As metrics we use BLEU and NTER.

- BLEU = $BP \times (\Pi prec_n)^{1/4}$. BP is brevity penality. $prec_n$ is precision of n-gram matches.
- RIBES = $(\tau + 1)/2 \times prec_1^{1/4}$, with Kendall's τ computed by measuring permutation between matching words in reference and hypothesis[5].
- NTER=$\max(1-\text{TER}, 0)$, which normalizes Translation Edit Rate[6] so that NTER=1 is best.

We compare two multi-objective approaches:

1. Linear-Combination of metrics (Eq. 2), optimized with PRO. We search a range of combination settings: $(p_1, p_2) = \{(0, 1), (0.3, 0.7), (0.5, 0.5), (0.7, 0.3), (1, 0)\}$. Note $(1, 0)$ reduces to standard single-metric optimization of e.g. BLEU.
2. Proposed Pareto approach (PMO-PRO).

Evaluation of multi-objective problems can be tricky because there is no single figure-of-merit. We thus adopted the following methodology: We run both methods 5 times (i.e. using the 5 different (p_1, p_2) setting each time) and $I = 20$ iterations each. For each method, this generates 5x20=100 results, and we plot the Pareto Frontier of these points in a 2-dimensional metric space (e.g. see Figure 2). A method is deemed better if its final Pareto Frontier curve is strictly dominating the other. We report devset results here; testset trends are similar but not included due to space constraints.[7]

[4] A difference with traditional MERT is the necessity of sentence-BLEU (Liang et al., 2006) in line 6. We use sentence-BLEU for optimization but corpus-BLEU for evaluation here.

[5] from `www.kecl.ntt.co.jp/icl/lirg/ribes`

[6] from `www.umd.edu/~snover/tercom`

[7] An aside: For comparing optimization methods, we believe devset comparison is preferable to testset since data mismatch may confound results. If one worries about generalization, we advocate to *re-decode* the devset with final weights and evaluate its 1-best output (which is done here). This is preferable to simply reporting the achieved scores on devset N-best (as done in some open-source scripts) since the learned weight may pick out good hypotheses in the N-best but perform poorly when re-decoding the same devset. The *re-decode devset* approach avoids being overly optimistic while accurately measuring optimization performance.

	Train	Devset	#Feat	Metrics
PubMed	0.2M	2k	14	BLEU, RIBES
NIST	7M	1.6k	8	BLEU, NTER

Table 1: Task characteristics: #sentences in Train/Dev, # of features, and metrics used. Our MT models are trained with standard phrase-based Moses software (Koehn and others, 2007), with IBM M4 alignments, 4gram SRILM, lexical ordering for PubMed and distance ordering for the NIST system. The decoder generates 50-best lists each iteration. We use SVMRank (Joachims, 2006) as optimization subroutine for PRO, which efficiently handle all pairwise samples without the need for sampling.

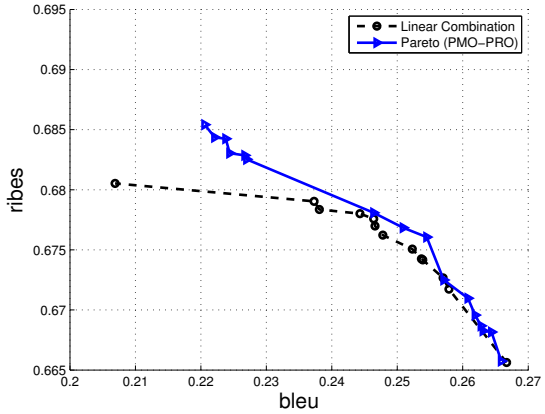

Figure 2: PubMed Results. The curve represents the Pareto Frontier of all results collected after multiple runs.

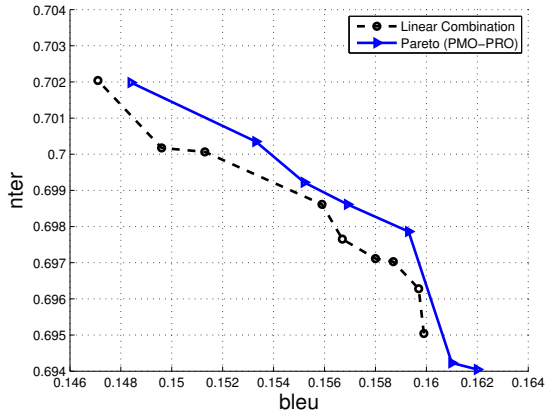

Figure 3: NIST Results

4.2 Results

Figures 2 and 3 show the results for PubMed and NIST, respectively. A method is better if its Pareto Frontier lies more towards the upper-right hand corner of the graph. Our observations are:

1. PMO-PRO generally outperforms Linear-Combination with any setting of (p_1, p_2). The Pareto Frontier of PMO-PRO dominates that of Linear-Combination. This implies PMO is effective in optimizing towards Pareto hypotheses.

2. For both methods, trading-off between metrics is necessary. For example in PubMed, the designer would need to make a choice between picking the best weight according to BLEU (BLEU=.265,RIBES=.665) vs. another weight with higher RIBES but poorer BLEU, e.g. (.255,.675). Nevertheless, both the PMO and Linear-Combination with various (p_1, p_2) samples this joint-objective space broadly.

3. Interestingly, a multi-objective approach can sometimes outperform a single-objective optimizer in its own metric. In Figure 2, single-objective PRO focusing on optimizing RIBES only achieves 0.68, but PMO-PRO using both BLEU and RIBES outperforms with 0.685.

The third observation relates to the issue of *metric tunability* (Liu et al., 2011). We found that RIBES can be difficult to tune directly. It is an extremely non-smooth objective with many local optima–slight changes in word ordering causes large changes in RIBES. So the best way to improve RIBES is to not to optimize it directly, but jointly with a more tunable metric BLEU. The learning curve in Figure 4 show that single-objective optimization of RIBES quickly falls into local optimum (at iteration 3) whereas PMO can zigzag and sacrifice RIBES in intermediate iterations (e.g. iteration 2, 15) leading to a stronger result ultimately. The reason is the diversity of solutions provided by the Pareto Frontier. This finding suggests that multi-objective approaches may be preferred, especially when dealing with new metrics that may be difficult to tune.

4.3 Additional Analysis and Discussions

What is the training time? The Pareto approach does not add much overhead to PMO-PRO. While FindParetoFrontier scales quadratically by size of N-best list, Figure 5 shows that the runtime is triv-

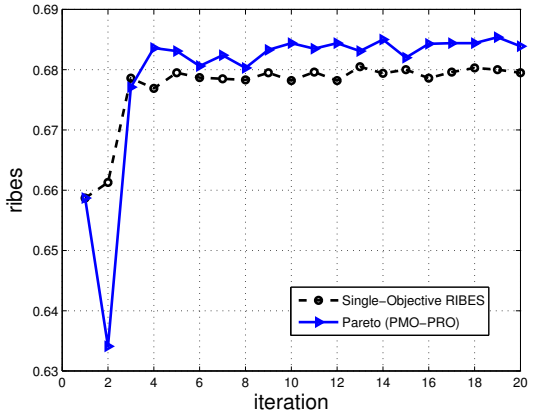

Figure 4: Learning Curve on RIBES: comparing single-objective optimization and PMO.

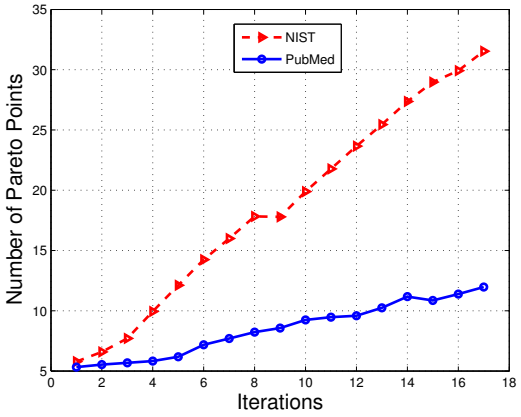

Figure 6: Average number of Pareto points

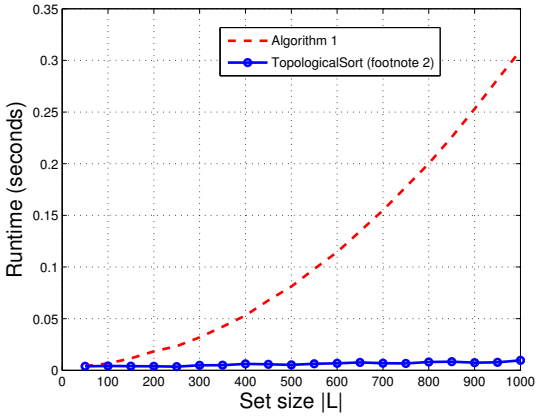

Figure 5: Avg. runtime per sentence of FindPareto

ial (0.3 seconds for 1000-best). Table 2 shows the time usage breakdown in different iterations for PubMed. We see it is mostly dominated by decoding time (constant per iteration at 40 minutes on single 3.33GHz processor). At later iterations, Opt takes more time due to larger file I/O in SVMRank. Note Decode and Pareto can be "embarrasingly parallelized."

Iter	Time	Decode (line 5)	Pareto (line 7)	Opt (line 10)	Misc. (line 6,8)
1	47m	85%	1%	1%	13%
10	62m	67%	6%	8%	19%
20	91m	47%	15%	22%	16%

Table 2: Training time usage in PMO-PRO (Algo 2).

How many Pareto points? The number of pareto hypotheses gives a rough indication of the diversity of hypotheses that can be exploited by PMO. Figure 6 shows that this number increases gradually per iteration. This perhaps gives PMO-PRO more directions for optimizing around potential local optimal. Nevertheless, we note that tens of Pareto points is far few compared to the large size of N-best lists used at later iterations of PMO-PRO. This may explain why the differences between methods in Figure 3 are not more substantial. Theoretically, the number will eventually level off as it gets increasingly harder to generate new Pareto points in a crowded space (Bentley et al., 1978).

Practical recommendation: We present the Pareto approach as a way to agnostically optimize multiple metrics jointly. However, in practice, one may have intuitions about metric tradeoffs even if one cannot specify $\{p_k\}$. For example, we might believe that approximately 1-point BLEU degradation is acceptable only if RIBES improves by at least 3-points. In this case, we recommend the following trick: Set up a multi-objective problem where one metric is BLEU and the other is $3/4$BLEU+$1/4$RIBES. This encourages PMO to explore the joint metric space but avoid solutions that sacrifice too much BLEU, and should also outperform Linear Combination that searches only on the $(3/4,1/4)$ direction.

5 Related Work

Multi-objective optimization for MT is a relatively new area. Linear-combination of BLEU/TER is

the most common technique (Zaidan, 2009), sometimes achieving good results in evaluation campaigns (Dyer et al., 2009). As far as we known, the only work that directly proposes a multi-objective technique is (He and Way, 2009), which modifies MERT to optimize a single metric subject to the constraint that it does not degrade others. These approaches all require some setting of constraint strength or combination weights $\{p_k\}$. Recent work in MT evaluation has examined combining metrics using machine learning for better correlation with human judgments (Liu and Gildea, 2007; Albrecht and Hwa, 2007; Gimnez and Màrquez, 2008) and may give insights for setting $\{p_k\}$. We view our Pareto-based approach as orthogonal to these efforts.

The tunability of metrics is a problem that is gaining recognition (Liu et al., 2011). If a good evaluation metric could not be used for tuning, it would be a pity. The Tunable Metrics task at WMT2011 concluded that BLEU is still the easiest to tune (Callison-Burch et al., 2011). (Mauser et al., 2008; Cer et al., 2010) report similar observations, in addition citing WER being difficult and BLEU-TER being amenable. One unsolved question is whether metric tunability is a problem inherent to the metric only, or depends also on the underlying optimization algorithm. Our positive results with PMO suggest that the choice of optimization algorithm can help.

Multi-objective ideas are being explored in other NLP areas. (Spitkovsky et al., 2011) describe a technique that alternates between hard and soft EM objectives in order to achieve better local optimum in grammar induction. (Hall et al., 2011) investigates joint optimization of a supervised parsing objective and some extrinsic objectives based on downstream applications. (Agarwal et al., 2011) considers using multiple signals (of varying quality) from online users to train recommendation models. (Eisner and Daumé III, 2011) trades off speed and accuracy of a parser with reinforcement learning. None of the techniques in NLP use Pareto concepts, however.

6 Opportunities and Limitations

We introduce a new approach (PMO) for training MT systems on multiple metrics. Leveraging the diverse perspectives of different evaluation metrics has the potential to improve overall quality. Based on Pareto Optimality, PMO is easy to implement and achieves better solutions compared to linear-combination baselines, for *any setting* of combination weights. Further we observe that multi-objective approaches can be helpful for optimizing difficult-to-tune metrics; this is beneficial for quickly introducing new metrics developed in MT evaluation into MT optimization, especially when good $\{p_k\}$ are not yet known. We conclude by drawing attention to some limitations and opportunities raised by this work:

Limitations: (1) The performance of PMO is limited by the size of the Pareto set. Small N-best lists lead to sparsely-sampled Pareto Frontiers, and a much better approach would be to enlarge the hypothesis space using lattices (Macherey et al., 2008). How to compute Pareto points directly from lattices is an interesting open research question. (2) The binary distinction between pareto vs. non-pareto points ignores the fact that 2nd-place non-pareto points may also lead to good practical solutions. A better approach may be to adopt a *graded* definition of Pareto optimality as done in some multi-objective works (Deb et al., 2002). (3) A robust evaluation methodology that enables significance testing for multi-objective problems is sorely needed. This will make it possible to compare multi-objective methods on more than 2 metrics. We also need to follow up with human evaluation.

Opportunities: (1) There is still much we do not understand about metric tunability; we can learn much by looking at joint metric-spaces and examining how new metrics correlate with established ones. (2) Pareto is just one approach among many in multi-objective optimization. A wealth of methods are available (Marler and Arora, 2004) and more experimentation in this space will definitely lead to new insights. (3) Finally, it would be interesting to explore other creative uses of multiple-objectives in MT beyond multiple metrics. For example: Can we learn to translate faster while sacrificing little on accuracy? Can we learn to jointly optimize cascaded systems, such as as speech translation or pivot translation? Life is full of multiple competing objectives.

Acknowledgments

We thank the reviewers for insightful feedback.

References

Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 132–140, New York, NY, USA. ACM.

J. Albrecht and R. Hwa. 2007. A re-examination of machine learning approaches for sentence-level mt evaluation. In *ACL*.

J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. 1978. On the average number of maxima in a set of vectors and applications. *Journal of the Association for Computing Machinery (JACM)*, 25(4).

Alexandra Birch, Phil Blunsom, and Miles Osborne. 2010. Metrics for MT evaluation: Evaluating reordering. *Machine Translation*, 24(1).

S. Börzsönyi, D. Kossmann, and K. Stocker. 2001. The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering (ICDE)*.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July. Association for Computational Linguistics.

Daniel Cer, Christopher Manning, and Daniel Jurafsky. 2010. The best lexical metric for phrase-based statistical MT system optimization. In *NAACL HLT*.

David Chiang, Wei Wang, and Kevin Knight. 2009. 11,001 new features for statistical machine translation. In *NAACL*.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passiveaggressive algorithms. *Journal of Machine Learning Research*, 7.

Kalyanmoy Deb, Amrit Pratap, Sammer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2).

Chris Dyer, Hendra Setiawan, Yuval Marton, and Philip Resnik. 2009. The university of maryland statistical machine translation system for the fourth workshop on machine translation. In *Proc. of the Fourth Workshop on Machine Translation*.

Jason Eisner and Hal Daumé III. 2011. Learning speed-accuracy tradeoffs in nondeterministic inference algorithms. In *COST: NIPS 2011 Workshop on Computational Trade-offs in Statistical Learning*.

Jesús Gimnez and Lluís Màrquez. 2008. Heterogeneous automatic mt evaluation through non-parametric metric combinations. In *ICJNLP*.

Parke Godfrey, Ryan Shipley, and Jarek Gyrz. 2007. Algorithms and analyses for maximal vector computation. *VLDB Journal*, 16.

Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the ntcir-9 workshop. In *Proceedings of the NTCIR-9 Workshop Meeting*.

Keith Hall, Ryan McDonald, Jason Katz-Brown, and Michael Ringgaard. 2011. Training dependency parsers by jointly optimizing multiple objectives. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1499, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Yifan He and Andy Way. 2009. Improving the objective function in minimum error rate training. In *MT Summit*.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

H. Isozaki, T. Hirao, K. Duh, K. Sudoh, and H. Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *EMNLP*.

T. Joachims. 2006. Training linear SVMs in linear time. In *KDD*.

P. Koehn et al. 2007. Moses: open source toolkit for statistical machine translation. In *ACL*.

A. Lavie and A. Agarwal. 2007. METEOR: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Workshop on Statistical Machine Translation*.

P. Liang, A. Bouchard-Cote, D. Klein, and B. Taskar. 2006. An end-to-end discriminative approach to machine translation. In *ACL*.

Ding Liu and Daniel Gildea. 2007. Source-language features and maximum correlation training for machine translation evaluation. In *NAACL*.

Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *EMNLP*.

R. T. Marler and J. S. Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26.

Arne Mauser, Saša Hasan, and Hermann Ney. 2008. Automatic evaluation measures for statistical machine

translation system optimization. In *International Conference on Language Resources and Evaluation*, Marrakech, Morocco, May.

Kaisa Miettinen. 1998. *Nonlinear Multiobjective Optimization*. Springer.

J.A. Nelder and R. Mead. 1965. The downhill simplex method. *Computer Journal*, 7(308).

Franz Och. 2003. Minimum error rate training in statistical machine translation. In *ACL*.

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2007. Labelled dependencies in machine translation evaluation. In *Proceedings of the Second Workshop on Statistical Machine Translation*.

Sebastian Pado, Daniel Cer, Michel Galley, Dan Jurafsky, and Christopher D. Manning. 2009. Measuring machine translation quality as semantic equivalence: A metric based on entailment features. *Machine Translation*, 23(2-3).

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

Vilfredo Pareto. 1906. *Manuale di Economica Politica, (Translated into English by A.S. Schwier as Manual of Political Economy, 1971)*. Societa Editrice Libraria, Milan.

Michael Paul. 2010. Overview of the iwslt 2010 evaluation campaign. In *IWSLT*.

Yoshikazu Sawaragi, Hirotaka Nakayama, and Tetsuzo Tanino, editors. 1985. *Theory of Multiobjective Optimization*. Academic Press.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *AMTA*.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011. Lateen em: Unsupervised training with multiple objectives, applied to dependency grammar induction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1269–1280, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Omar Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. In *The Prague Bulletin of Mathematical Linguistics*.