FrugalScore: Learning Cheaper, Lighter and Faster Evaluation Metrics for Automatic Text Generation

Anonymous ACL submission

Abstract

Fast and reliable evaluation metrics are key to R&D progress. While traditional natural language generation metrics are fast, they are not very reliable. Conversely, new metrics based on large pretrained language models are much more reliable, but require significant computational resources. In this paper, we propose FrugalScore, an approach to learn a fixed, low cost version of any expensive NLG metric, while retaining most of its original performance. Experiments with BERTScore and MoverScore on summarization and translation show that FrugalScore is on par with the original metrics (and sometimes better), while having several orders of magnitude less parameters and running several times faster. On average over all learned metrics, tasks, and variants, FrugalScore retains 96.8% of the performance, runs 24 times faster, and has 35 times less parameters than the original metrics. We make our trained metrics publicly available [1], to benefit the entire NLP community and in particular researchers and practitioners with limited resources.

1 Introduction

Automatic evaluation metrics are the only way to monitor the training of, evaluate, and compare across models in a systematic, large-scale way, and are thus a critical component of the research and development ecosystem in machine learning. To get adopted in practice, evaluation metrics need to be both reliable and affordable, i.e., fast and easy to compute.

While some metrics meet these criteria, such as precision and recall in information retrieval, root mean square error in regression, etc., finding suitable metrics is still an open problem in the field of Natural Language Generation (NLG) (Novikova et al., 2017).

Indeed, historical n-gram matching metrics such as ROUGE (Lin, 2004) for summarization, BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) for translation, while affordable, are not very reliable, as they are based on surface-form matching only, i.e., lexical similarity, and have thus no sense of semantic similarity. For instance, it makes little sense to use ROUGE for the evaluation of abstractive summarization systems (which are becoming the norm), or whenever the generated text paraphrases the original text.

Following the advent of transfer learning in NLP, new NLG metrics based on large pretrained language models have recently been proposed, such as BERTScore (Zhang et al., 2019) and MoverScore (Zhao et al., 2019). By relying on contextual embeddings, these metrics capture semantics and are therefore much more reliable. However, due to the sheer size of the underlying models, these metrics pose environmental issues (Strubell et al., 2019), take time to compute, and require access to significant computational resources, so they are not accessible by everyone in the NLP community.

For example, we were not able to run some of the best variants of BERTScore[2], based on DeBERTa-Large and DeBERTa-XLarge (He et al., 2020) on a 12GB GPU. Even when enough GPU memory is available, relying on such large models is still associated with extended runtimes, which can impede the progress of experiments when used once or more per epoch for validation and monitoring purposes.

To address this problem, we propose in this paper FrugalScore, an approach to learn a lightweight version of BERTScore, MoverScore, and more generally any metric based on a large pretrained language model.

Our contributions can be summarized as follows: 1) Our compact models have several orders of magnitude less parameters than the original metrics and run several times faster, while retaining most of their original performance. We even outperform

[1] Link will be provided upon acceptance.

[2] From BERTScore's authors: https://tinyurl.com/8cwyter2

the original metrics in some cases[3].
2) Our metrics are not only faster because of the much smaller amount of parameters, but also because they do not rely on any similarity function.
3) Regardless of how expensive the original metric is, querying our trained metrics always has the same low, fixed cost. This decoupling is a major advantage as the size of the pretrained language models has recently been growing tremendously (e.g., Brown et al. (2020)).

2 Background

Related work falls into two categories: unsupervised and supervised metrics.

2.1 Unsupervised metrics

To address the limitations of ROUGE and BLEU, variants based on static word embeddings (Mikolov et al., 2013) were developed, e.g., ROUGE-WE (Ng and Abrecht, 2015), BLEU2VEC (Tättar and Fishel, 2017), and MEANT 2.0 (Lo, 2017). While using word vectors is a progress over strict n-gram matching, static embeddings are still very limited as they do not capture polysemy, i.e., the fact that words have different meanings in different contexts.

More recently, the focus has shifted to harnessing the power of the contextualized embeddings produced by large pretrained language models. For instance, the Sentence Mover's Similarity (Clark et al., 2019) represents sentences as the average of their ELMo word embeddings (Peters et al., 2018) and measures the minimum cost of transforming one summary into the other, using a modified version of the Word Mover's Distance (Kusner et al., 2015). BERTR (Mathur et al., 2019) computes approximate recall based on the pairwise cosine similarity between the BERT embeddings (Devlin et al., 2018) of the words in automatic and reference translations. Mark-Evaluate (Mordido and Meinel, 2020) is a family of metrics that consider contextualized word or sentence embeddings derived from BERT as population samples, to evaluate language generation with population estimation methods used in ecology.

Finally, the recently introduced BERTScore (Zhang et al., 2019) and MoverScore (Zhao et al., 2019) are general-purpose NLG evaluation metrics that are becoming widely used. The main difference between BERTScore and MoverScore lies in the function used to compute the similarity between the representations of the two sequences $\mathbf{x} = \langle \mathbf{x_1}, ..., \mathbf{x_k} \rangle$ and $\mathbf{y} = \langle \mathbf{y_1}, ..., \mathbf{y_l} \rangle$. We experimented with these two metrics, so we provide more details about them in what follows.

BERTScore first computes the pairwise cosine similarity between the representations of the tokens in each sequence, and uses greedy matching to match each token to the most similar one in the other sequence. Given two pre-normalized vector sequences $\mathbf{x}$ and $\mathbf{y}$, BERTScore computes:

$$R_{BERT} = \frac{1}{|\mathbf{x}|} \sum_{\mathbf{x_i} \in x} \max_{\mathbf{y}_j \in \mathbf{y}} \mathbf{x_i^T} \mathbf{y}_j \tag{1}$$

and:

$$P_{BERT} = \frac{1}{|\mathbf{y}|} \sum_{\mathbf{y_i} \in y} \max_{\mathbf{x}_j \in \mathbf{x}} \mathbf{y_i^T} \mathbf{x}_j \tag{2}$$

The F1-score is classically obtained as:

$$F_{BERT} = 2 \frac{P_{BERT} R_{BERT}}{P_{BERT} + R_{BERT}} \tag{3}$$

MoverScore uses an n-gram generalization of the Word Mover's Distance (WMD) (Kusner et al., 2015) as their (dis)similarity function. More specifically, they solve for the optimal transportation flow matrix $F \in R^{|\mathbf{x}| \times |\mathbf{y}|}$ between the two weighted sequences of n-grams:

$$WMD(\mathbf{x}, \mathbf{y}) = min_F \langle C, F \rangle \tag{4}$$
$$s.t. \quad F\mathbf{1} = f_{\mathbf{x}}, \quad F^T \mathbf{1} = f_{\mathbf{y}}$$

Where C is the transportation cost matrix (C_{ij} is the Euclidean distance between x_i and y_j) and $f_x \in R_+^{|\mathbf{x}|}$ and $f_y \in R_+^{|\mathbf{y}|}$ are the n-gram weight vectors.

Note that by directly learning BERTScore's and MoverScore's full internal mapping (from sequence pairs to final scalar scores), FrugalScore internalizes their similarity functions. This does not only provide a speedup at inference time, but also improves performance, as shown in section 5.

2.2 Supervised metrics

Related to our work are also supervised metrics, which are directly trained on human evaluations. ROSE (Conroy and Dang, 2008) is a linear combination model of different variants of ROUGE using canonical correlation. BEER (Stanojević and

[3]Hence the name FrugalScore, as frugal engineering is defined as "achieving more with fewer resources".

Sima'an, 2014) is a learning-to-rank approach using word and character n-gram matching, and token ordering, as features to maximize correlation with human rankings of machine translation systems. S^3 (Peyrard et al., 2017) trains a regression model that takes the evaluation scores of several existing metrics and many hand-crafted features as input, and learns the best combination of them to approximate human summary judgments. DPMFcomb (Yu et al., 2015) and Blend (Ma et al., 2017) are combined metrics incorporating a vast amount of lexical, syntactic and semantic based translation evaluation metrics using ranking and regression SVMs respectively. RUSE (Shimanaka et al., 2018) evaluates machine translation with a neural regressor based on universal sentence embeddings (e.g., InferSent (Conneau et al., 2017)). NUBIA (Kane et al., 2020) consists of three modules: a feature extractor based on RoBERTa (Liu et al., 2019) and GPT-2 (Radford et al., 2019) fine-tuned on language evaluation tasks, an aggregator trained to predict the quality of the hypothesis given the reference using the extracted features, and a calibrator mapping all predictions between 0 and 1.

Differences. Like the aforementioned efforts, FrugalScore is a learned metric. However, it does not rely on any intermediate or handcrafted features, and, most importantly, it does not require training on human annotations. Supervision in FrugalScore is conducted on a synthetic dataset, as a trick to expose and learn the internal mapping of the unsupervised metrics to be learned. Last but not least, unlike all aforementioned methods, compression is central to FrugalScore, which is based on miniature versions of the models used by the original metrics.

2.3 Knowledge distillation

Knowledge distillation (KD) (Hinton et al., 2015) is the process of transferring knowledge from a large teacher model to a smaller student model to accomplish model compression (Buciluǎ et al., 2006). It was originally proposed in the domain of computer vision and speech recognition, then successfully adapted to NLP (Sanh et al., 2019). Distillation can be accomplished in three ways: (1) offline, where a teacher is first pre-trained, then a student is trained under the guidance of the teacher (Hinton et al., 2015); (2) online, where the student and the teacher are trained simultaneously (Zhang et al., 2018); and (3) self, where the same model plays the role of student and teacher, e.g., transferring the knowledge of a later exit layer into an earlier one of the same multi-exit network (Phuong and Lampert, 2019). Previous studies on KD mainly focused on classification problems (Gou et al., 2021). A few attempts have been made on regression problems (Chen et al., 2017; Saputra et al., 2019; Takamoto et al., 2020), in which special losses were proposed to train the student with respect to both the teacher's regression outputs and ground truth scores. Different from conventional distillation, our work is more similar to *data-free* KD (Kang and Kang, 2021), where the student is trained in the absence of the dataset used to train the teacher. To transfer knowledge, we first create a synthetic dataset by annotating sequence pairs with a large model (teacher), and then train a miniature model (student) on that dataset, in an offline and regression setting.

2.4 Differences with BLEURT

A work closely related to ours is BLEURT (Sellam et al., 2020). However, there are a number of significant differences with our approach. First, BLEURT continues the pretraining of an already pretrained BERT-based model on a synthetic dataset in a self-supervised way, whereas FrugalScore is directly trained to learn the scores of the metric of interest, in a supervised fashion.

Also, BLEURT's synthetic dataset is made by perturbing Wikipedia sentences with mask-filling, backtranslation, and word dropping, whereas we use other data sources than Wikipedia such as summarization and translation datasets, and only NLG models to induce perturbations.

When creating its synthetic dataset, BLEURT automatically annotates the (original, perturbed) sequence pairs with numerical and categorical "signals": BLEU, ROUGE, BERTscore, backtranslation likelihood, textual entailment (probability of three labels: entail, contradict, and neutral, given by BERT fine-tuned on MNLI), and backtranslation flag. On the other hand, FrugalScore simply and directly annotates the sequence pairs with the metric to be learned.

After pretraining, BLEURT is fine-tuned on human judgments, in a way similar to the supervised metrics described in subsection 2.2. BLEURT does not learn to generate a scalar until that final fine-tuning phase, so it cannot be used as a metric before that. Conversely, FrugalScore is trained from the start to be a metric, and the fine-tuning phase is optional.

Also, BLEURT was designed for the evaluation of translation. The authors only test whether it can be applied to a different task by experimenting on the WebNLG (data-to-text) dataset (Gardent et al., 2017). Conversely, we focus on learning general text similarity metrics (e.g., BERTscore and MoverScore), so FrugalScore is task-agnostic by design.

Finally, and above all, the objective of FrugalScore is model compression, whereas that of BLEURT is metric learning.

3 Our Approach

Developing FrugalScore requires three phases, one of which is optional.

Phase 1. We create a synthetic dataset (see subsection 3.1) by sampling pairs of more or less related sequences and annotating them with the expensive metrics to be learned. This is a one-time operation that does not need to be repeated regardless of the model used in Phase 2.

Phase 2. We continue the pretraining (subsection 3.2) of a miniature pretrained language model on the synthetic dataset built by Phase 1. Here, the miniature model learns the internal mapping of the expensive metric, including any similarity function applied to the representations. Note that a different miniature is trained for each metric to be learned (we leave learning metric combinations as future work).

The miniature can then be used in inference mode to generate scores for any never-seen pair of sequences.

Phase 3 (optional). We fine-tune the miniature on human annotations, which, as shown in section 6, can boost performance.

3.1 Synthetic dataset

The objective here was to generate pairs of sequences mimicking the (reference, candidate) pairs found in NLG datasets, which are usually semantically related and in many cases paraphrasing one another. We sampled our sequences from a variety of data sources, listed next.

Summarization. For each document in the well-known CNN/DailyMail dataset (Nallapati et al., 2016), our goal was to generate several summaries differing in terms of structure and quality. To this purpose, we used different pretrained seq2seq summarization models: BART-base and BART-large (Lewis et al., 2019), mBART (Liu et al., 2020), and BARThez (Kamal Eddine et al., 2021). BART is a seq2seq autoencoder with a Transformer architecture.

The four models were fine-tuned for one epoch on 50k examples randomly sampled from the training set of CNN/DM, and were used to generate summaries for the whole training set of 287,112 documents, using greedy decoding.

Note that we kept the 50K documents used for fine-tuning in the final generation pool, in order to create quality differences among summaries. Indeed, models are expected to better summarize the documents used for training than never-seen documents.

We also used the human reference summaries, so that in the end, each document was associated with 5 summaries, resulting in 10 pairs of summaries per document.

Backtranslation. We also generated paraphrases with backtranslation, by sampling sentences from the OpenSubtitle English monolingual corpus (Lison and Tiedemann, 2016), and translating them to French, Arabic and German with OPUS-MT (Tiedemann and Thottingal, 2020), before translating them back to English. We used OPUS-MT because of its ready-to-use checkpoints available for many language pairs. We ended up with 4 variations for each sentence (including the original one), resulting in 6 paraphrase pairs per sentence.

Denoising. To avoid bias towards summarization and translation, we also generated pairs of related sequences such that the first element in the pair was a Wikipedia segment and the second element was a BART-denoised version of it (Lewis et al., 2019).

More precisely, we sampled 2M segments from Wikipedia such that the number of unigrams in these segments was uniformly distributed between 1 and 200. Our assumption was that enforcing variations in sequence length would help the learned metric to generalize.

We then applied BART's *text infilling* and *sentence permutation* perturbation strategies to each segment. That is, multiple text spans were sampled and replaced with a `[MASK]` special token. The lengths of the spans were sampled from a Poisson distribution ($\lambda = 3$). 50% of the tokens within the input segment were masked and 20% of the masked text was replaced with random tokens (creating pathological examples to increase the robustness of the learned metric). The sentences in the

input segment were then shuffled.

We finally used a BART-Base checkpoint[4] from the Fairseq library (Ott et al., 2019) to try to reconstruct the perturbed versions of the original sequences, hence creating variants of them.

Annotating pairs. We sampled 4.5M sequence pairs uniformly from each aforelisted source. These pairs were then annotated with the metrics to be learned. Note that this is a one-time operation that does not need to be repeated regardless of which models are trained downstream.

In this work, we experimented with two recent expensive NLG metrics that rely on large pretrained language models, BERTScore (Zhang et al., 2019) and MoverScore (Zhao et al., 2019), presented in section 2. However, it is important to note that our method can be used with any other NLG metric.

Note that for BERTScore, we used the F-1 score F_{BERT}, as recommended by the authors (Zhang et al., 2019). For MoverScore, still following the authors (Zhao et al., 2019), we used the variant operating on unigrams and the IDF to compute the vectors of weights.

3.2 Metric learning

We continue the pretraining of three BERT miniatures[5] on our synthetic dataset: BERT-Tiny ($L = 2$, $H = 128$), BERT-Small ($L = 4$, $H = 512$) and BERT-Medium ($L = 8$, $H = 512$), where L is the number of layers and H is the dimension of the embedding space. These models have respectively 25 times, 3.78 times, and 2.64 times less parameters than BERT-base. The concept of BERT miniatures was introduced by Turc et al. (2019) to test whether pretraining small models from scratch was competitive to distilling very large models. The miniature models have already been pretrained on masked language model and next sentence prediction objectives.

We continue pretraining using the standard method introduced by Devlin et al. (2018). We concatenate the two sequences $x = \langle x_1, ..., x_k \rangle$ and $y = \langle y_1, ..., y_l \rangle$ in a given pair, separating them with a special `[SEP]` token. A special `[CLS]` token is also added at the beginning of the resulting sequence. The sequence of contextualised embeddings $\langle \mathbf{z_{[CLS]}}, \mathbf{x_1}, ...\mathbf{x_k}, \mathbf{z_{[SEP]}}, \mathbf{y_1}, ..., \mathbf{y_l} \rangle$ is then obtained. We finally add a fully connected layer on top, that linearly projects the $\mathbf{z_{[CLS]}}$ vector to a scalar s.

The model is trained to minimize the mean square error (MSE) loss between the learned metric s_i and the metric to be learned $\hat{s_i}$ (i.e., the annotation of the pair):

$$l = \frac{1}{N} \sum_{n=1}^{N} ||s_i - \hat{s_i}||^2 \quad (5)$$

When pretraining is over, the models can be further fine-tuned on smaller human-annotated datasets as shown in section 6, or directly used to generate scores for unseen examples as shown in section 4.

Setup. We use a batch size of 32 and the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 3×10^{-5}, linear decay, and a warm-up for 6% of the total training steps, and we train the model for three epochs. We conducted the pretraining on a single TITAN RTX GPU (24GB). It took 10, 24 and 33 hours, respectively for the tiny, small, and medium miniatures. We rely on the Transformers library (Wolf et al., 2019) for all pretraining and fine-tuning experiments.

4 Experiments

In this section, FrugalScore is used in inference mode to generate scores directly after pretraining, i.e., no fine-tuning is performed (see section 6 for fine-tuning results).

We evaluate on two text generation tasks: summarization and translation. We use evaluation datasets containing (reference, candidate) sequence pairs annotated with human scores assessing the quality of the candidates given the references. We measure the effectiveness of FrugalScore by measuring the Pearson correlation of its scores with the human judgments and comparing it to that of the original metrics. We also take the number of parameters and the runtime into account.

Text summarization. We use 4 different multi-document summarization datasets from the Text Analysis Conference (TAC)[6]: TAC-2008, TAC-2009, TAC-2010 and TAC-2011.

These datasets respectively contain 48, 44, 46 and 44 clusters of documents and 58, 55, 43 and 51 systems are used to generate summaries. Each cluster forms a topic to be summarized and has 4 reference summaries. There are approximately 10k pairs in each dataset. Each pair is annotated with

[4] https://dl.fbaipublicfiles.com/fairseq/models/bart.base.tar.gz
[5] https://huggingface.co/google
[6] https://tac.nist.gov/

	Metric	Model	Scores (TAC)	Runtime (TAC)	Scores (WMT)	Runtime (WMT)	Params
a	BERTScore	BERT-Tiny	55.4/47.5	1m 27s	37.6	1m 22s	4.4M
b	BERTScore	BERT-Small	61.6/51.5	2m 20s	39.1	1m 42s	29.1M
c	BERTScore	BERT-Medium	62.7/52.4	2m 28s	39.8	2m 04s	41.7M
d	BERTScore	BERT-Base	64.7/54.7	3m 28s	41.9	2m 09s	110M
e	BERTScore	RoBERTa-Large	64.2/55.4	5m 17s	43.2	3m 03s	355M
f	BERTScore	DeBERTa-XLarge	64.5/**56.0**	6m 20s	**44.5**	3m 49s	900M
g	MoverScore	BERT-Base	**66.5**/55.4	301m 29s	44.0	64m 32s	110M
i	FrugalScore$_d$	BERT-Tiny	64.9/53.5	1m 28s	38.4	1m 18s	4.4M
ii	FrugalScore$_d$	BERT-Small	64.7/53.7	2m 29s	41.3	1m 35s	29.1M
iii	FrugalScore$_d$	BERT-Medium	64.8/54.2	3m 41s	41.9	1m 55s	41.7M
iv	FrugalScore$_e$	BERT-Tiny	60.0/50.1	1m 28s	37.5	1m 18s	4.4M
v	FrugalScore$_e$	BERT-Small	64.1/53.8	2m 29s	40.5	1m 35s	29.1M
vi	FrugalScore$_e$	BERT-Medium	63.9/52.1	3m 41s	41.7	1m 55s	41.7M
vii	FrugalScore$_f$	BERT-Tiny	61.7/51.0	1m 28s	38.0	1m 18s	4.4M
viii	FrugalScore$_f$	BERT-Small	66.0/54.9	2m 29s	41.5	1m 35s	29.1M
ix	FrugalScore$_f$	BERT-Medium	65.5/54.9	3m 41s	43.0	1m 55s	41.7M
x	FrugalScore$_g$	BERT-Tiny	**67.3/55.1**	1m 28s	39.8	1m 18s	4.4M
xi	FrugalScore$_g$	BERT-Small	65.9/54.7	2m 29s	42.8	1m 35s	29.1M
xii	FrugalScore$_g$	BERT-Medium	66.2/**55.1**	3m 41s	**43.6**	1m 55s	41.7M

Table 1: Scores are summary-level (TAC) and segment-level (WMT) Pearson correlations averaged over 2008 to 2011 for TAC (pyramid score/responsiveness) and over all source languages for WMT-2019. Runtimes include preprocessing. Subscripts refer to row labels and indicate which metric-model combination was used to annotate pairs (e.g., for FrugalScore$_d$, it is row d, i.e., BERTScore-BERT-Base).

two human judgment scores: the *Pyramid Score* (Harnly et al., 2005) and the *Responsiveness* (Dang et al., 2008). The former measures the proportion of important semantic units (SCUs) in the reference summaries captured by the system summary, while the latter reflects the content coverage and the readability of each summary.

Machine translation. Our evaluation corpus is from the WMT-2019[7] shared task (Li et al., 2019). We consider all the to-English pairs: Chinese, Czech, German, Finnish, Russian, Lithuanian and Kazakh to English. For each language, we use the test set that contains several thousands of reference-candidate pairs annotated with human ratings that assess the translation quality.

5 Results

Table 1 reports the results averaged over the 4 TAC datsets and the 7 WMT to-English language pairs. Details are provided in Appendices A and B.

We benchmarked the metrics in terms of Pearson correlations with human scores, runtimes, and numbers of parameters. We used two approaches to compute the Pearson correlations: summary-level (or segment-level) and system-level.

In the former approach, a score is attributed to each of the output candidates, while in the latter approach, one single overall score is attributed to the system (by averaging its individual scores).

Rows `a` to `c` correspond to BERTScore with BERT miniatures as the underlying model. They are simple baselines added for the sake of comparison, to see what we get when BERTScore is used with the same number of parameters as FrugalScore.

Rows `d` to `g` correspond to the expensive metrics that are learned by FrugalScore (in the respective sections of the bottom half of the table). They are BERTScore and MoverScore metrics where the underlying model is a large pretrained language model: BERT-Base ($L = 12$, $H = 512$), RoBERTa-Large ($L = 24$, $H = 1024$) (Liu et al., 2019), and DeBERTa-XLarge ($L = 24$, $H = 1536$) (He et al., 2020).

Finally, rows `i` to `xii` correspond to FrugalScore. Subscripts refer to row labels and indicate which metric-model combination was used to annotate pairs. I.e., FrugalScore$_d$ learned the metric of row d, i.e., BERTScore with BERT-Base.

First, results show that all FrugalScores, regardless of which metric they learned, significantly out-

[7] http://www.statmt.org/wmt19/

perform the BERTScores with miniature models. These results suggest that FrugalScore is a better approach than using an existing metric with an already-compressed underlying model. The reason for this is probably that in FrugalScore, the knowledge of the original unsupervised metric (based on a large model) is explicitly transferred to the miniature via the continuation of its pretraining on the synthetic dataset. That is, the miniature is actually learning a metric. Whereas, on the other hand, plugging a compressed version of a general-purpose language model into the original unsupervised metric just makes it lose expressiveness and capacity.

Second, we can clearly see that FrugalScore retains most of the performance of the original metric, while running several times faster and reducing the number of parameters by several orders of magnitude. On average over all metrics, tasks, and miniatures, FrugalScore retains 96.8% of the original performance, runs 24 times faster, and has 35 times less parameters.

More precisely, on average across all metrics, FrugalScore-Tiny retains 97.7/94.7% of the original performance on TAC (pyramid score/responsiveness), while running 54 times faster and having 84 times less parameters. Its small and medium versions retain near full performance in terms of responsiveness (98 and 97.7%) and even slightly outperform the original metrics in terms of pyramid score, while at the same time reducing the runtime and the number of parameters by 32 (resp. 21) and 13 (resp. 9) times.

On WMT, FrugalScore-Tiny retains 88.58% of the performance of the original metrics, while running 14 times faster (and still having 84 times less parameters), while the small and medium versions of FrugalScore retain 95.71 % and 98.06% of the original performance while still offering a 32 times (resp. 21) speedup and having 13 times (resp. 9) less parameters, on average.

Interestingly, FrugalScore even improves the performance of the original metrics in some cases. For example, on TAC, FrugalScore$_g$ with BERT-Tiny (row x) improves the performance of the original MoverScore metric based on BERT-Base (row g) from 66.5 to 67.3 in terms of pyramid score, while reducing the number of parameters by 25 and running 50 times faster. Other examples, also for TAC with the pyramid score, include FrugalScore$_f$ with BERT-Small (row viii, +1.5 point) and FrugalScore$_f$ with BERT-medium (row ix, +1 point).

Finally, the results of FrugalScore for different miniature sizes show that, on WMT, using larger models always improves performance (e.g., row x $\rightarrow$ xi $\rightarrow$ xii). But interestingly, on TAC, this observation does not hold (e.g., row vi $\rightarrow$ viii $\rightarrow$ ix), and sometimes, FrugalScore with the smallest miniature (BERT-Tiny) is superior (e.g. rows i and x). This finding suggests that the impact of the pretrained language model size is task-dependent.

To sum up, results clearly show the effectiveness of FrugalScore in learning a cheaper, lighter, and faster version of the original metrics, while retaining most of their original performance. The system-level correlations, provided in Appendices C and D, corroborate these positive results.

We also provide the correlations between the original and the learned metrics in Appendices E and F. It is interesting to note that a greater correlation with the original metric is not always associated with a better performance. E.g., the tiny version of FrugalScore$_g$ is the best (row x), while it is the less correlated with the original metric.

6 Fine-tuning on Human Annotations

We test two hypotheses in this section: (1) whether fine-tuning on a human-annotated dataset is beneficial, and (2) when fine-tuning on human annotations, whether continuing pretraining on our synthetic dataset is useful.

Because we cannot use the same dataset for fine-tuning and evaluation, we fine-tune a BERT-Small on each year of TAC 2008-2011 for 4 epochs, using two other years as the validation set, and the remaining year as the test set. The best epoch is selected based on validation performance. We use a batch size of 32 and a learning rate of 2e-5 that linearly decreases to zero. Finally, we experiment with two scenarios: fine-tuning the miniature directly without continuing its pretraining on our synthetic dataset, and fine-tuning it after the pretraining continuation (with annotations generated by BERTScore-BERT-Base).

Results. Results are reported in Table 2 in terms of summary-level Pearson correlations with human evaluations (Pyramid), averaged over 3 runs with different random seeds.

First, it is obvious that everywhere, continuing the pretraining on our synthetic dataset leads to a significant boost in performance. This is in accor-

	Pretraining Continued	TAC-2008	TAC-2009	TAC-2010	TAC-2011	Average
TAC-2008	no	-	$67.7_{0.57}$	$66.1_{0.18}$	$63.6_{0.36}$	65.8
	yes		$74.4_{0.13}$	$71.3_{0.04}$	$67.3_{0.13}$	71.0
TAC-2009	no	$61.4_{0.41}$	-	$66.9_{0.24}$	$62.7_{0.55}$	63.7
	yes	$65.8_{0.25}$		$70.7_{0.32}$	$66.0_{0.18}$	67.5
TAC-2010	no	$59.7_{0.47}$	$67.3_{0.7}$	-	$62.4_{0.47}$	63.1
	yes	$64.7_{0.19}$	$74.3_{0.24}$		$67.2_{0.11}$	68.7
TAC-2011	no	$57.6_{1.39}$	$64.7_{1.03}$	$66.5_{0.66}$	-	62.9
	yes	$63.9_{0.31}$	$72.0_{0.44}$	$71.6_{0.44}$		69.2

Table 2: Summary-level Pearson correlations with human judgments (Pyramid scores), averaged over 3 runs (standard deviation as subscript). Rows correspond to the training sets and columns to the test sets.

dance with Sellam et al. (2020), who found that pretraining was beneficial even in a supervised setting.

Second, even if a direct comparison is not possible, we can remark when looking at the TAC Pyramid score of row ii) in Table 1 (FrugalScore$_d$-BERT-Small) that fine-tuning after pretraining seems very beneficial too. Indeed, after fine-tuning, we reach on average 71, 67.5, 68.7, and 69.2 (depending on the split), which represents overall a gain of 4.4 points over the non-fine-tuned model (score of 64.7).

7 Impact of Data Sources

To test the importance of each data source introduced in subsection 3.1, we created a training set containing sequence pairs uniformly and equally sampled from each source. We annotated these pairs with the BERTScore-BERT-Base metric and we used them to continue the pretraining of a BERT-Small miniature.

We also considered pairs drawn at random from the pairs generated with the other strategies. The motivation for random pairs was to sample "negative examples", as seeing only "positive examples" (pairs of related sequences) could bias the learned metric towards considering any two unrelated sequences as similar.

We then continued the pretraining of the BERT-Small miniature four times, excluding each time the pairs coming from a specific data source. We evaluated the learned metric on TAC-2008 to 2011 and on WMT-2019. Figure 1 shows the average improvements in the Pearson correlation with human judgments relative to training a model on all sources. Note that when training on all four sources, we sampled 30k pairs from each source (120k total), and when excluding a source, we sampled 40k pairs from each source (120k total).

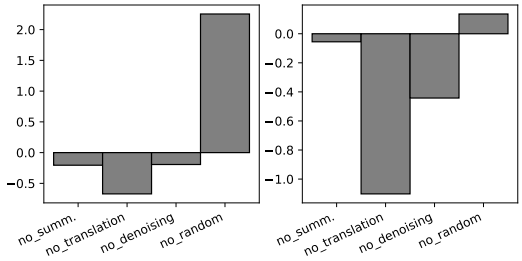

Figure 1: Relative improvement in Pearson correlation compared to a dataset covering all sources. Left: TAC. Right: WMT.

We can clearly see that excluding the random pairs improves performance while excluding any of the other data sources decreases performance. In other words, all our data sources are beneficial, and it is not necessary to add "negative examples". We hypothesise that this is due to the fact that NLG datasets typically do not contain completely unrelated pairs of sentences. Interestingly, the pairs generated with the backtranslation strategy have the greatest impact on performance.

8 Conclusion

We proposed FrugalScore, an approach to learn a fixed, low-cost version of any expensive NLG evaluation metric. Experiments on summarization and translation tasks show that our FrugalScore versions of BERTScore and MoverScore retain most of the original performance in terms of the correlation with human judgments, while running several times faster and having several orders of magnitude less parameters. On average over all learned metrics, tasks, and variants, FrugalScore retains 96.8% of the performance, runs 24 times faster, and has 35 times less parameters than the original metrics.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. 2017. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30.

Elizabeth Clark, Asli Celikyilmaz, and Noah A Smith. 2019. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

John M. Conroy and Hoa Trang Dang. 2008. Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 145–152, Manchester, UK. Coling 2008 Organizing Committee.

Hoa Trang Dang, Karolina Owczarzak, et al. 2008. Overview of the tac 2008 update summarization task. In *TAC*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Aaron Harnly, Ani Nenkova, Rebecca Passonneau, and Owen Rambow. 2005. Automation of summary evaluation by the pyramid method. In *Recent Advances in Natural Language Processing (RANLP)*, pages 226–232.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Moussa Kamal Eddine, Antoine Tixier, and Michalis Vazirgiannis. 2021. BARThez: a skilled pretrained French sequence-to-sequence model. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9369–9390, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hassan Kane, Muhammed Yusuf Kocyigit, Ali Abdalla, Pelkins Ajanoh, and Mohamed Coulibali. 2020. NUBIA: NeUral based interchangeability assessor for text generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 28–37, Online (Dublin, Ireland). Association for Computational Linguistics.

Myeonginn Kang and Seokho Kang. 2021. Data-free knowledge distillation in neural networks for regression. *Expert Systems with Applications*, 175:114813.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xian Li, Paul Michel, Antonios Anastasopoulos, Yonatan Belinkov, Nadir Durrani, Orhan Firat, Philipp Koehn, Graham Neubig, Juan Pino, and Hassan Sajjad. 2019. Findings of the first shared task on machine translation robustness. *arXiv preprint arXiv:1906.11943*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chi-kiu Lo. 2017. MEANT 2.0: Accurate semantic MT evaluation for any output language. In *Proceedings of the Second Conference on Machine Translation*, pages 589–597, Copenhagen, Denmark. Association for Computational Linguistics.

Qingsong Ma, Yvette Graham, Shugen Wang, and Qun Liu. 2017. Blend: a novel combined mt metric based on direct assessment—casict-dcu submission to wmt17 metrics task. In *Proceedings of the second conference on machine translation*, pages 598–603.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2019. Putting evaluation in context: Contextual embeddings improve machine translation evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2799–2808.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Gonçalo Mordido and Christoph Meinel. 2020. Mark-evaluate: Assessing language generation using population estimation methods. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1963–1977, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1925–1930, Lisbon, Portugal. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Mary Phuong and Christoph H Lampert. 2019. Distillation-based training for multi-exit architectures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1355–1364.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Muhamad Risqi U Saputra, Pedro PB De Gusmao, Yasin Almalioglu, Andrew Markham, and Niki Trigoni. 2019. Distilling knowledge from a deep pose regressor network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 263–272.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*.

Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2018. RUSE: Regressor using sentence embeddings for automatic machine translation evaluation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 751–758, Belgium, Brussels. Association for Computational Linguistics.

Miloš Stanojević and Khalil Sima'an. 2014. Beer: Better evaluation as ranking. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 414–419.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Makoto Takamoto, Yusuke Morishita, and Hitoshi Imaoka. 2020. An efficient method of training small models for regression problems with knowledge distillation. In *2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 67–72. IEEE.

Andre Tättar and Mark Fishel. 2017. bleu2vec: the painfully familiar metric on continuous vector space steroids. In *Proceedings of the Second Conference on Machine Translation*, pages 619–622.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Hui Yu, Qingsong Ma, Xiaofeng Wu, and Qun Liu. 2015. CASICT-DCU participation in WMT2015 metrics task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 417–421, Lisbon, Portugal. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*.

A Detailed TAC Evaluation per Year

	Metric	Model	TAC-2008	TAC-2009	TAC-2010	TAC-2011	Macro Avg. Score	Runtime	Params
a	BERTScore	BERT-Tiny	52.1/44.4	62.2/51.9	54.6/49.9	52.7/43.6	55.4/47.5	1m 27s	4.4M
b	BERTScore	BERT-Small	56.0/47.8	70.0/54.6	61.1/54.5	59.1/49.2	61.6/51.5	2m 20s	29.1M
c	BERTScore	BERT-Medium	57.3/48.5	70.6/55.3	63.1/56.2	59.7/49.5	62.7/52.4	2m 28s	41.7M
d	BERTScore	BERT-Base	61.3/52.2	73.2/58.7	63.3/56.8	61.0/51.2	64.7/54.7	3m 28s	110M
e	BERTScore	RoBERTa-Large	56.4/50.9	71.1/58.3	**69.1/61.4**	60.3/50.8	64.2/55.4	5m 17s	355M
f	BERTScore	DeBERTa-XLarge	60.9/**54.5**	**73.9/60.4**	62.6/56.0	61.5/**53.0**	64.5/**56.0**	6m 20s	900M
g	MoverScore	BERT-Base	**64.7**/54.2	**73.9**/58.2	64.7/57.0	**62.6**/52.5	**66.5**/55.4	301m 29s	110M
i	FrugalScore$_d$	BERT-Tiny	60.9/50.0	72.5/56.4	64.8/57.5	61.4/50.0	64.9/53.5	1m 28s	4.4M
ii	FrugalScore$_d$	BERT-Small	61.9/51.8	73.0/57.3	62.6/55.8	61.3/50.0	64.7/53.7	1m 35s	29.1M
iii	FrugalScore$_d$	BERT-Medium	62.0/52.2	73.3/**58.1**	62.6/56.0	61.3/50.6	64.8/54.2	1m 55s	41.7M
iv	FrugalScore$_e$	BERT-Tiny	54.8/46.4	66.8/54.2	61.8/53.1	56.4/46.7	60.0/50.1	1m 28s	4.4M
v	FrugalScore$_e$	BERT-Small	59.1/49.6	72.7/55.7	**68.1**/59.8	63.0/50.1	64.1/53.8	2m 29s	29.1M
vi	FrugalScore$_e$	BERT-Medium	57.9/48.4	71.8/54.4	65.7/57.0	60.3/48.5	63.9/52.1	3m 41s	41.7M
vii	FrugalScore$_f$	BERT-Tiny	57.8/48.5	68.6/55.7	63.0/54.8	57.5/47.8	61.7/51.0	1m 28s	4.4M
viii	FrugalScore$_f$	BERT-Small	60.1/51.0	73.5/57.5	67.3/59.5	63.1/51.7	66.0/54.9	2m 29s	29.1M
ix	FrugalScore$_f$	BERT-Medium	59.0/50.3	73.3/57.4	67.2/**60.2**	62.4/51.5	65.5/54.9	3m 41s	41.7M
x	FrugalScore$_g$	BERT-Tiny	63.6/51.7	**74.4**/57.3	68.0/60.1	**63.2**/51.2	**67.3/55.1**	1m 28s	4.4M
xi	FrugalScore$_g$	BERT-Small	63.2/52.5	73.1/57.1	65.1/57.6	62.3/51.5	65.9/54.7	2m 29s	29.1M
xii	FrugalScore$_g$	BERT-Medium	**63.8/53.2**	73.6/57.7	65.3/57.5	62.1/**51.8**	66.2/**55.1**	3m 41s	41.7M

Table 3: Summary-level Pearson correlation (pyramid score/responsiveness).

B Detailed WMT Evaluation per Language

	Metric	Model	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Macro Avg. Score	Runtime	Params
a	BERTScore	BERT-Tiny	29.7	32.5	33.9	52.0	40.5	30.7	44.2	37.6	1m 22s	4.4M
b	BERTScore	BERT-Small	30.0	33.6	34.6	52.4	42.3	31.8	49.1	39.1	1m 42s	29.1M
c	BERTScore	BERT-Medium	30.8	34.4	35.2	52.8	42.8	32.4	50.3	39.8	2m 04s	41.7M
d	BERTScore	BERT-Base	32.8	37.4	37.1	54.0	44.7	33.7	53.7	41.9	2m 09s	110M
e	BERTScore	RoBERTa-Large	35.3	38.7	38.7	52.0	45.3	34.3	**58.3**	43.2	3m 03s	355M
f	BERTScore	DeBERTa-XLarge	**37.6**	**39.2**	**40.3**	53.4	**47.3**	**35.7**	57.8	**44.5**	3m 49s	900M
g	MoverScore	BERT-Base	36.5	39.1	39.3	**55.0**	46.5	35.6	56.0	44.0	64m 32s	110M
i	FrugalScore$_d$	BERT-Tiny	30.2	32.8	34.6	52.4	39.9	31.2	47.7	38.4	1m 18s	4.4M
ii	FrugalScore$_d$	BERT-Small	32.6	35.9	37.1	54.1	43.5	33.6	52.3	41.3	1m 35s	29.1M
iii	FrugalScore$_d$	BERT-Medium	32.9	37.0	37.4	54.4	44.3	34.1	53.2	41.9	1m 55s	41.7M
iv	FrugalScore$_e$	BERT-Tiny	30.6	32.8	33.0	49.8	38.7	29.8	48.1	37.5	1m 18s	4.4M
v	FrugalScore$_e$	BERT-Small	33.7	35.4	35.4	51.6	42.6	32.6	52.5	40.5	1m 35s	29.1M
vi	FrugalScore$_e$	BERT-Medium	35.2	37.1	35.6	52.0	44.0	33.8	54.4	41.7	1m 55s	41.7M
vii	FrugalScore$_f$	BERT-Tiny	30.8	33.1	34.4	50.8	39.4	30.4	47.1	38.0	1m 18s	4.4M
viii	FrugalScore$_f$	BERT-Small	34.5	36.4	37.0	52.7	43.9	33.4	52.6	41.5	1m 35s	29.1M
ix	FrugalScore$_f$	BERT-Medium	35.8	**38.3**	37.7	53.4	45.7	34.8	**55.1**	43.0	1m 55s	41.7M
x	FrugalScore$_g$	BERT-Tiny	33.0	34.0	36.2	53.6	40.5	32.7	48.6	39.8	1m 18s	4.4M
xi	FrugalScore$_g$	BERT-Small	35.6	37.4	38.9	55.0	44.8	34.8	52.8	42.8	1m 35s	29.1M
xii	FrugalScore$_g$	BERT-Medium	**36.2**	**38.3**	**39.1**	**55.6**	**45.8**	**35.3**	54.7	**43.6**	1m 55s	41.7M

Table 4: Segment-level Pearson correlation.

C Detailed TAC Evaluation per Year (System Level)

	Metric	Model	TAC-2008	TAC-2009	TAC-2010	TAC-2011	Macro Avg. Score	Runtime	Params
a	BERTScore	BERT-Tiny	82.5/77.6	87.4/81.8	77.5/75.0	82.1/79.2	82.4/78.4	1m 27s	4.4M
b	BERTScore	BERT-Small	84.4/81.4	95.8/84.0	81.3/78.0	87.6/85.3	87.3/82.2	2m 20s	29.1M
c	BERTScore	BERT-Medium	86.3/82.7	96.0/84.6	84.0/80.6	87.8/85.5	88.5/83.3	2m 28s	41.7M
d	BERTScore	BERT-Base	90.6/87.5	96.5/87.5	83.7/80.9	88.3/86.4	89.8/85.6	3m 28s	110M
e	BERTScore	RoBERTa-Large	80.0/80.9	94.7/87.7	**92.7/89.8**	88.9/89.2	89.1/86.9	5m 17s	355M
f	BERTScore	DeBERTa-XLarge	88.0/**89.8**	**97.5/89.8**	85.7/84.0	**90.7/91.8**	90.5/**88.9**	6m 20s	900M
g	MoverScore	BERT-Base	**95.4**/89.5	96.9/85.9	85.7/84.0	88.6/86.0	**91.7**/86.3	301m 29s	110M
i	FrugalScore$_d$	BERT-Tiny	91.6/85.3	95.8/84.7	86.2/82.9	88.3/84.4	90.5/84.3	1m 28s	4.4M
ii	FrugalScore$_d$	BERT-Small	90.9/86.8	96.2/85.4	82.8/79.6	87.8/84.3	89.4/84.0	1m 35s	29.1M
iii	FrugalScore$_d$	BERT-Medium	90.6/87.0	96.6/86.3	82.5/79.6	87.6/84.9	89.3/84.5	1m 55s	41.7M
iv	FrugalScore$_e$	BERT-Tiny	86.3/81.1	95.1/87.1	84.5/80.2	84.5/80.9	87.6/82.3	1m 28s	4.4M
v	FrugalScore$_e$	BERT-Small	85.1/81.7	95.7/83.6	**91.2**/87.5	91.7/87.5	90.9/85.1	2m 29s	29.1M
vi	FrugalScore$_e$	BERT-Medium	81.6/80.7	95.7/84.1	90.9/87.5	87.6/85.3	89.0/84.4	3m 41s	41.7M
vii	FrugalScore$_f$	BERT-Tiny	89.7/84.5	95.3/**87.6**	85.1/81.4	84.8/81.2	88.7/83.7	1m 28s	4.4M
viii	FrugalScore$_f$	BERT-Small	86.8/85.1	96.7/85.4	89.5/86.2	91.6/88.7	91.2/86.3	2m 29s	29.1M
ix	FrugalScore$_f$	BERT-Medium	85.4/86.3	**97.2**/87.2	91.1/**88.9**	**92.3/91.0**	91.5/**88.3**	3m 41s	41.7M
x	FrugalScore$_g$	BERT-Tiny	**93.7**/86.1	96.2/83.9	90.1/87	89.4/84.8	**92.3**/85.5	1m 28s	4.4M
xi	FrugalScore$_g$	BERT-Small	93.2/**87.6**	96.4/84.2	85/81.7	87.9/84.9	90.6/84.6	2m 29s	29.1M
xii	FrugalScore$_g$	BERT-Medium	**93.7**/87.5	96.5/84.5	84.8/81.6	87.3/84.7	90.6/84.6	3m 41s	41.7M

Table 5: System-level Pearson correlation (pyramid/responsiveness).

D Detailed WMT Evaluation per Language (System Level)

	Metric	Model	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Macro Avg. Score	Runtime	Params
a	BERTScore	BERT-Tiny	74.1	97.9	93.1	99.77	87.9	94.5	91.7	91.3	1m 22s	4.4M
b	BERTScore	BERT-Small	82.6	97.5	88.2	**99.87**	95.3	96.4	93.0	93.3	1m 42s	29.1M
c	BERTScore	BERT-Medium	83.7	97.7	88.2	99.86	94.4	96.2	93.5	93.4	2m 04s	41.7M
d	BERTScore	BERT-Base	89.1	97.8	89.7	99.72	96.9	96.9	95.8	95.1	2m 09s	110M
e	BERTScore	RoBERTa-Large	**94.0**	98.4	98.1	98.00	96.1	91.0	98.2	96.3	3m 03s	355M
f	BERTScore	DeBERTa-XLarge	93.9	98.3	**98.2**	99.18	**98.7**	97.1	**98.4**	**97.7**	3m 49s	900M
g	MoverScore	BERT-Base	88.1	**99.1**	91.2	98.58	96.0	**97.2**	96.4	95.2	64m 32s	110M
i	FrugalScore$_d$	BERT-Tiny	81.1	98.6	94.4	99.80	92.2	95.4	93.8	93.6	1m 18s	4.4M
ii	FrugalScore$_d$	BERT-Small	86.5	98.5	93.6	99.82	95.9	97.1	94.7	95.2	1m 35s	29.1M
iii	FrugalScore$_d$	BERT-Medium	88.3	98.3	92.1	99.79	96.4	97.2	95.4	95.4	1m 55s	41.7M
iv	FrugalScore$_e$	BERT-Tiny	80.2	97.7	94.9	99.73	86.4	94.6	93.7	92.5	1m 18s	4.4M
v	FrugalScore$_e$	BERT-Small	83.9	98.0	95.2	99.79	92.4	97.0	95.1	94.5	1m 35s	29.1M
vi	FrugalScore$_e$	BERT-Medium	88.1	97.9	93.0	99.78	94.9	**97.8**	96.1	95.4	1m 55s	41.7M
vii	FrugalScore$_f$	BERT-Tiny	81.3	97.9	96.1	99.81	89.8	94.7	93.7	93.3	1m 18s	4.4M
viii	FrugalScore$_f$	BERT-Small	85.8	97.7	**96.2**	**99.85**	95.3	97.3	95.7	95.4	1m 35s	29.1M
ix	FrugalScore$_f$	BERT-Medium	**89.9**	97.9	90.8	**99.85**	**97.6**	**97.8**	**96.9**	**95.8**	1m 55s	41.7M
x	FrugalScore$_g$	BERT-Tiny	81.8	**98.9**	95.6	99.73	92.1	95.6	94.4	94.0	1m 18s	4.4M
xi	FrugalScore$_g$	BERT-Small	85.4	98.8	95.8	99.52	94.9	96.8	95.3	95.2	1m 35s	29.1M
xii	FrugalScore$_g$	BERT-Medium	87.0	98.8	93.5	99.29	95.6	97.0	95.9	95.3	1m 55s	41.7M

Table 6: System-level Pearson correlation.

E Correlation with Learned Metric (TAC)

	Metric	Model	TAC-2008	TAC-2009	TAC-2010	TAC-2011	Average
i	FrugalScore$_d$	BERT-Tiny	91.7	94.7	97.2	95.1	94.7
ii	FrugalScore$_d$	BERT-Small	96.9	97.9	99.0	98.0	98.0
iii	FrugalScore$_d$	BERT-Medium	98.3	98.8	99.4	99.0	98.9
iv	FrugalScore$_e$	BERT-Tiny	77.9	82.4	87.5	75.9	80.9
v	FrugalScore$_e$	BERT-Small	86.9	90.7	91.6	89.2	89.6
vi	FrugalScore$_e$	BERT-Medium	87.1	90.7	86.3	90.9	88.8
vii	FrugalScore$_f$	BERT-Tiny	80.0	85.5	89.4	81.3	84.0
viii	FrugalScore$_f$	BERT-Small	88.9	92.8	92.6	91.4	91.4
ix	FrugalScore$_f$	BERT-Medium	89.9	92.9	92.1	93.6	92.1
x	FrugalScore$_g$	BERT-Tiny	91.1	94.8	95.7	94.8	94.1
xi	FrugalScore$_g$	BERT-Small	94.8	97.4	98.4	98.0	97.1
xii	FrugalScore$_g$	BERT-Medium	96.4	98.0	98.9	98.6	98.0

Table 7: Summary-level Pearson correlation between the FrugalScore$_{d,e,f,g}$ and the metrics d, e, f, g used to generate the annotations.

F Correlation with Learned Metric (WMT)

	Metric	Model	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Average
i	FrugalScore$_d$	BERT-Tiny	90.2	89.6	91.3	92.5	92.4	92.4	92.3	91.5
ii	FrugalScore$_d$	BERT-Small	96.3	96.1	96.8	97.2	97.2	97.3	97.3	96.9
iii	FrugalScore$_d$	BERT-Medium	97.5	97.5	98.0	98.2	98.3	98.3	98.3	98.0
iv	FrugalScore$_e$	BERT-Tiny	71.0	74.6	78.6	82.0	82.1	82.0	81.8	78.9
v	FrugalScore$_e$	BERT-Small	81.4	83.7	84.7	86.9	87.1	87.1	87.2	85.4
vi	FrugalScore$_e$	BERT-Medium	85.0	87.6	87.5	89.2	89.5	89.5	89.6	88.3
vii	FrugalScore$_f$	BERT-Tiny	71.6	76.4	81.3	83.9	83.7	83.6	83.5	80.6
viii	FrugalScore$_f$	BERT-Small	82.2	85.5	88.6	90.1	90.1	90.1	90.1	88.1
ix	FrugalScore$_f$	BERT-Medium	85.9	89.4	91.7	92.7	92.7	92.7	92.7	91.1
x	FrugalScore$_g$	BERT-Tiny	89.3	89.1	90.8	91.7	91.6	91.8	91.5	90.8
xi	FrugalScore$_g$	BERT-Small	94.5	94.7	95.7	96.1	96.0	96.1	95.9	95.6
xii	FrugalScore$_g$	BERT-Medium	95.7	96.1	96.9	97.2	97.1	97.1	97.0	96.7

Table 8: Segment-level Pearson correlation between the FrugalScore$_{d,e,f,g}$ and the metrics d, e, f, g used to generate the annotations.