SHEF-Lite 2.0: Sparse Multi-task Gaussian Processes for Translation Quality Estimation

Daniel Beck and **Kashif Shah** and **Lucia Specia**
Department of Computer Science
University of Sheffield
Sheffield, United Kingdom
{debeck1,kashif.shah,l.specia}@sheffield.ac.uk

Abstract

We describe our systems for the WMT14 Shared Task on Quality Estimation (subtasks 1.1, 1.2 and 1.3). Our submissions use the framework of Multi-task Gaussian Processes, where we combine multiple datasets in a multi-task setting. Due to the large size of our datasets we also experiment with Sparse Gaussian Processes, which aim to speed up training and prediction by providing sensible sparse approximations.

1 Introduction

The purpose of machine translation (MT) quality estimation (QE) is to provide a quality prediction for new, unseen machine translated texts, without relying on reference translations (Blatz et al., 2004; Specia et al., 2009; Bojar et al., 2013). A common use of quality predictions is the decision between post-editing a given machine translated sentence and translating its source from scratch, based on whether its post-editing effort is estimated to be lower than the effort of translating the source sentence.

The WMT 2014 QE shared task defined a group of tasks related to QE. In this paper, we describe our submissions for subtasks 1.1, 1.2 and 1.3. Our models are based on Gaussian Processes (GPs) (Rasmussen and Williams, 2006), a non-parametric kernelised probabilistic framework. We propose to combine multiple datasets to improve our QE models by applying GPs in a multi-task setting. Our hypothesis is that using sensible multi-task learning settings gives improvements over simply pooling all datasets together.

Task 1.1 focuses on predicting post-editing effort for four language pairs: English-Spanish (**en-es**), Spanish-English (**es-en**), English-German (**en-de**), and German-English (**de-en**). Each contains a different number of source sentences and their human translations, as well as 2-3 versions of machine translations: by a statistical (SMT) system, a rule-based system (RBMT) system and, for en-es/de only, a hybrid system. Source sentences were extracted from tests sets of WMT13 and WMT12, and the translations were produced by top MT systems of each type and a human translator. Labels range from 1 to 3, with 1 indicating a perfect translation and 3, a low quality translation.

The purpose of task 1.2 is to predict HTER scores (Human Translation Error Rate) (Snover et al., 2006) using a dataset composed of 896 English-Spanish sentences translated by a MT system and post-edited by a professional translator. Finally, task 1.3 aims at predicting post-editing time, using a subset of 650 sentences from the Task 1.2 dataset.

For each task, participants can submit two types of results: scoring and ranking. For scoring, evaluation is made in terms of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). For ranking, DeltaAvg and Spearman's rank correlation were used as evaluation metrics.

2 Model

Gaussian Processes are a Bayesian non-parametric machine learning framework considered the state-of-the-art for regression. They assume the presence of a latent function $f : \mathbb{R}^F \rightarrow \mathbb{R}$, which maps a vector $\mathbf{x}$ from feature space F to a scalar value. Formally, this function is drawn from a GP prior:

$$f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')),$$

which is parameterised by a mean function (here, $\mathbf{0}$) and a covariance kernel function $k(\mathbf{x}, \mathbf{x}')$. Each response value is then generated from the function evaluated at the corresponding input, $y_i = f(\mathbf{x}_i) + \eta$, where $\eta \sim \mathcal{N}(0, \sigma_n^2)$ is added white-noise.

Proceedings of the Ninth Workshop on Statistical Machine Translation, pages 307–312,
Baltimore, Maryland USA, June 26–27, 2014.

Prediction is formulated as a Bayesian inference under the posterior:

$$p(y_*|\mathbf{x}_*, \mathcal{D}) = \int_f p(y_*|\mathbf{x}_*, f)p(f|\mathcal{D}),$$

where $\mathbf{x}_*$ is a test input, y_* is the test response value and $\mathcal{D}$ is the training set. The predictive posterior can be solved analitically, resulting in:

$$y_* \sim \mathcal{N}(\mathbf{k}_*^T(\mathbf{K} + \sigma_n^2 I)^{-1}\mathbf{y}, \\ k(x_*, x_*) - \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2 I)^{-1}\mathbf{k}_*),$$

where $\mathbf{k}_* = [k(\mathbf{x}_*, \mathbf{x}_1)k(\mathbf{x}_*, \mathbf{x}_2) \dots k(\mathbf{x}_*, \mathbf{x}_n)]^T$ is the vector of kernel evaluations between the training set and the test input and $\mathbf{K}$ is the kernel matrix over the training inputs (the Gram matrix).

The kernel function encodes the covariance (similarity) between each input pair. While a variety of kernel functions are available, here we followed previous work in QE using GP (Cohn and Specia, 2013; Shah et al., 2013) and employed a squared exponential (SE) kernel with automatic relevance determination (ARD):

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{F}\frac{x_i - x_i'}{l_i}\right),$$

where F is the number of features, σ_f^2 is the covariance *magnitude* and $l_i > 0$ are the feature *lengthscales*.

The resulting model hyperparameters (SE variance σ_f^2, noise variance σ_n^2 and SE lengthscales l_i) were learned from data by maximising the model likelihood. All our models were trained using the GPy[1] toolkit, an open source implementation of GPs written in Python.

2.1 Multi-task learning

The GP regression framework can be extended to multiple outputs by assuming $f(\mathbf{x})$ to be a vector valued function. These models are commonly referred as *coregionalization* models in the GP literature (Álvarez et al., 2012). Here we refer to them as *multi-task* kernels, to emphasize our application.

In this work, we employ a separable multi-task kernel, similar to the one used by Bonilla et al. (2008) and Cohn and Specia (2013). Considering a set of D tasks, we define the corresponding multi-task kernel as:

$$k((\mathbf{x}, d), (\mathbf{x}', d')) = k_{\text{data}}(\mathbf{x}, \mathbf{x}') \times \mathbf{B}_{d,d'}, \quad (1)$$

where k_{data} is a kernel on the input points, d and d' are task or metadata information for each input and $\mathbf{B} \in \mathbb{R}^{D \times D}$ is the multi-task matrix, which encodes task covariances. For task 1.1, we consider each language pair as a different task, while for tasks 1.2 and 1.3 we use additional datasets for the same language pair (en-es), treating each dataset as a different task.

To perform the learning procedure the multi-task matrix should be parameterised in a sensible way. We follow the parameterisations proposed by Cohn and Specia (2013), which we briefly describe here:

Independent: $\mathbf{B} = \mathbf{I}$. In this setting each task is modelled independently. This is not strictly equivalent to independent model training because the tasks share the same data kernel (and the same hyperparameters);

Pooled: $\mathbf{B} = \mathbf{1}$. Here the task identity is ignored. This is equivalent to pooling all datasets in a single task model;

Combined: $\mathbf{B} = \mathbf{1} + \alpha\mathbf{I}$. This setting leverages between independent and pooled models. Here, $\alpha > 0$ is treated as an hyperparameter;

Combined+: $\mathbf{B} = \mathbf{1} + \text{diag}(\boldsymbol{\alpha})$. Same as "combined", but allowing one different α value per task.

2.2 Sparse Gaussian Processes

The performance bottleneck for GP models is the Gram matrix inversion, which is $O(n^3)$ for standard GPs, with n being the number of training instances. For multi-task settings this can be a potential issue because these models replicate the instances for each task and the resulting Gram matrix has dimensionality $nd \times nd$, where d is the number of tasks.

Sparse GPs tackle this problem by approximating the Gram matrix using only a subset of m *inducing inputs*. Without loss of generalisation, consider these m points as the first instances in the training data. We can then expand the Gram matrix in the following way:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{mm} & \mathbf{K}_{m(n-m)} \\ \mathbf{K}_{(n-m)m} & \mathbf{K}_{(n-m)(n-m)} \end{bmatrix}.$$

Following the notation in (Rasmussen and Williams, 2006), we refer $\mathbf{K}_{m(n-m)}$ as $\mathbf{K}_{mn}$ and

[1] http://sheffieldml.github.io/GPy/

its transpose as $\mathbf{K}_{nm}$. The block structure of $\mathbf{K}$ forms the basis of the so-called Nyström approximation:

$$\tilde{\mathbf{K}} = \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn}, \quad (2)$$

which results in the following predictive posterior:

$$\begin{aligned} y_* \sim \mathcal{N}(&\mathbf{k}_{m*}^T\tilde{\mathbf{G}}^{-1}\mathbf{K}_{mn}\mathbf{y}, \quad (3)\\ &k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_{m*}^T\mathbf{K}_{mm}^{-1}\mathbf{k}_{m*} + \\ &\sigma_n^2\mathbf{k}_{m*}^T\tilde{\mathbf{G}}^{-1}\mathbf{k}_{m*}), \end{aligned}$$

where $\tilde{\mathbf{G}} = \sigma_n^2\mathbf{K}_{mm} + \mathbf{K}_{mn}\mathbf{K}_{nm}$ and $\mathbf{k}_{m*}$ is the vector of kernel evaluations between test input $\mathbf{x}_*$ and the m inducing inputs. The resulting training complexity is $O(m^2 n)$.

The remaining question is how to choose the inducing inputs. We follow the approach of Snelson and Ghahramani (2006), which note that these inducing inputs do not need to be a subset of the training data. Their method considers each input as a hyperparameter, which is then optimised jointly with the kernel hyperparameters.

2.3 Features

For all tasks we used the QuEst framework (Specia et al., 2013) to extract a set of 80 black-box features as in Shah et al. (2013), for which we had all the necessary resources available. Examples of the features extracted include:

- N-gram-based features:
  - Number of tokens in source and target segments;
  - Language model (LM) probability of source and target segments;
  - Percentage of source 1–3-grams observed in different frequency quartiles of a large corpus of the source language;
  - Average number of translations per source word in the segment as given by IBM 1 model from a large parallel corpus of the language, with probabilities thresholded in different ways.
- POS-based features:
  - Ratio of percentage of nouns/verbs/etc in the source and target segments;
  - Ratio of punctuation symbols in source and target segments;
  - Percentage of direct object personal or possessive pronouns incorrectly translated.

For the full set of features we refer readers to QuEst website.[2]

To perform feature selection, we followed the approach used in Shah et al. (2013) and ranked the features according to their learned lengthscales (from the lowest to the highest). The lengthscale of a feature can be interpreted as the relevance of such feature for the model. Therefore, the outcome of a GP model using an ARD kernel can be viewed as a list of features ranked by relevance, and this information can be used for feature selection by discarding the lowest ranked (least useful) ones.

3 Preliminary Experiments

Our submissions are based on multi-task settings. For task 1.1, we consider each language pair as a different task, training one model for all pairs. For tasks 1.2 and 1.3, we used additional datasets and encoded each one as a different task (totalling 3 tasks):

WMT13: these are the datasets provided in last year's QE shared task (Bojar et al., 2013). We combined training and test sets, totalling $2,754$ sentences for HTER prediction and $1,003$ sentences for post-editing time prediction, both for English-Spanish.

EAMT11: this dataset is provided by Specia (2011) and is composed of $1,000$ English-Spanish sentences annotated in terms of HTER and post-editing time.

For each task we prepared two submissions: one trained on a standard GP with the full 80 features set and another one trained on a sparse GP with a subset of 40 features. The features were chosen by training a smaller model on a subset of 400 instances and following the procedure explained in Section 2.3 for feature selection, with a pre-define cutoff point on the number of features (40), based on previous experiments. The sparse models were trained using 400 inducing inputs.

To select an appropriate multi-task setting for our submissions we performed preliminary experiments using a 90%/10% split on the corresponding training set for each task. The resulting MAE scores are shown in Tables 1 and 2, for standard and sparse GPs, respectively. The boldface figures correspond to the settings we choose for the

[2] http://www.quest.dcs.shef.ac.uk/quest_files/features_blackbox

	Task 1.1				Task 1.2	Task 1.3
	en-es	es-en	en-de	de-en	en-es	en-es
Independent	**0.4905**	0.5325	**0.5962**	**0.5452**	0.2047	0.4486
Pooled	0.4957	0.5171	0.6012	0.5612	**0.2036**	0.8599
Combined	0.4939	**0.5162**	0.6007	0.5550	0.2321	0.7489
Combined+	0.4932	0.5182	0.5990	0.5514	0.2296	**0.4472**

Table 1: MAE results for preliminary experiments on standard GPs. Post-editing time scores for task 1.3 are shown on log time per word.

	Task 1.1				Task 1.2	Task 1.3
	en-es	es-en	en-de	de-en	en-es	en-es
Independent	0.5036	0.5274	0.6002	0.5532	0.3432	**0.3906**
Pooled	0.4890	**0.5131**	0.5927	0.5532	**0.1597**	0.6410
Combined	**0.4872**	0.5183	0.5871	**0.5451**	0.2871	0.6449
Combined+	0.4935	0.5255	**0.5864**	0.5458	0.1659	0.4040

Table 2: MAE results for preliminary experiments on sparse GPs. Post-editing time scores for task 1.3 are shown on log time per word.

official submissions, after re-training on the corresponding full training sets.

To check the speed-ups obtained from using sparse GPs, we measured wall clock times for training and prediction in Task 1.1 using the "Independent" multi-task setting. Table 3 shows the resulting times and the corresponding speed-ups when comparing to the standard GP. For comparison, we also trained a model using 200 inducing inputs, although we did not use the results of this model in our submissions.

	Time (secs)	Speed-up
Standard GP	12122	–
Sparse GP (m=400)	3376	3.59x
Sparse GP (m=200)	978	12.39x

Table 3: Wall clock times and speed-ups for GPs training and prediction: full versus sparse GPs.

4 Official Results and Discussion

Table 4 shows the results for Task 1.1. Using standard GPs we obtained improved results over the baseline for English-Spanish and English-German only, with particularly substantial improvements for English-Spanish, which also happens for sparse GPs. This may be related to the larger size of this dataset when compared to the others. Our results here are mostly inconclusive though and we plan to investigate this setting more in depth in the future. Specifically, due to the coarse behaviour of the labels, ordinal regression GP models (like the one proposed in (Chu et al., 2005)) could be useful for this task.

Results for Task 1.2 are shown in Table 5. The standard GP model performed unusually poorly when compared to the baseline or the sparse GP model. To investigate this, we inspected the resulting model hyperparameters. We found out that the noise σ_n^2 was optimised to a very low value, close to zero, which characterises overfitting. The same behaviour was not observed with the sparse model, even though it had a much higher number of hyperparameters to optimise, and was therefore more prone to overfitting. We plan to investigate this issue further but a possible cause could be bad starting values for the hyperparameters.

Table 6 shows results for Task 1.3. In this task, the standard GP model outperformed the baseline, with the sparse GP model following very closely. These figures represent significant improvements compared to our submission to the same task in last year's shared task (Beck et al., 2013), where we were not able to beat the baseline. The main differences between last year's and this year's models are the use of additional datasets and a higher number of features (25 vs. 40). The competitive results for the sparse GP models are very promising because they show we can combine multiple datasets to improve post-editing time prediction while employing a sparse model to cope with speed issues.

	en-es		es-en		en-de		de-en	
	Δ	ρ	Δ	ρ	Δ	ρ	Δ	ρ
Standard GP	**0.21**	**-0.33**	0.11	-0.15	**0.26**	**-0.36**	0.24	**-0.27**
Sparse GP	**0.17**	**0.27**	0.12	-0.17	0.23	-0.33	0.14	-0.17
Baseline	0.14	-0.22	0.12	-0.21	0.23	-0.34	0.21	-0.25

	en-es		es-en		en-de		de-en	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Standard GP	**0.49**	**0.63**	0.62	0.77	**0.63**	**0.74**	0.65	**0.77**
Sparse GP	0.54	0.69	0.54	0.69	0.64	**0.75**	0.66	0.79
Baseline	0.52	0.66	0.57	0.68	0.64	0.76	0.65	0.78

Table 4: Official results for task 1.1. The top table shows results for the ranking subtask (Δ: DeltaAvg; ρ: Spearman's correlation). The bottom table shows results for the scoring subtask.

	Ranking		Scoring	
	Δ	ρ	MAE	RMSE
Standard GP	0.72	0.09	18.15	23.41
Sparse GP	**7.69**	**0.43**	**15.04**	**18.38**
Baseline	5.08	0.31	15.23	19.48

Table 5: Official results for task 1.2.

	Ranking		Scoring	
	Δ	ρ	MAE	RMSE
Standard GP	**16.08**	**0.64**	**17.13**	**27.33**
Sparse GP	**16.33**	**0.63**	**17.42**	**27.35**
Baseline	14.71	0.57	21.49	34.28

Table 6: Official results for task 1.3.

5 Conclusions

We proposed a new setting for training QE models based on Multi-task Gaussian Processes. Our settings combined different datasets in a sensible way, by considering each dataset as a different task and learning task covariances. We also proposed to speed-up training and prediction times by employing sparse GPs, which becomes crucial in multi-task settings. The results obtained are specially promising in the post-editing time task, where we obtained the same results as with standard GPs and improved over our models from the last evaluation campaign.

In the future, we plan to employ our multi-task models in large-scale settings, like datasets annotated through crowdsourcing platforms. These datasets are usually labelled by dozens of annotators and multi-task GPs have proved an interesting framework for learning the annotation noise (Cohn and Specia, 2013). However, multiple tasks can easily make training and prediction times prohibitive, and thus another direction if work is to use recent advances in sparse GPs, like the one proposed by Hensman et al. (2013). We believe that the combination of these approaches could further improve the state-of-the-art performance in these tasks.

Acknowledgments

This work was supported by funding from CNPq/Brazil (No. 237999/2012-9, Daniel Beck) and from European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 296347 (QTLaunchPad).

References

Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. 2012. Kernels for Vector-Valued Functions: a Review. *Foundations and Trends in Machine Learning*, pages 1–37.

Daniel Beck, Kashif Shah, Trevor Cohn, and Lucia Specia. 2013. SHEF-Lite : When Less is More for Translation Quality Estimation. In *Proceedings of WMT13*, pages 337–342.

John Blatz, Erin Fitzgerald, and George Foster. 2004. Confidence estimation for machine translation. In *Proceedings of the 20th Conference on Computational Linguistics*, pages 315–321.

Ondej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of WMT13*, pages 1–44.

Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K. I. Williams. 2008. Multi-task Gaussian Process Prediction. *Advances in Neural Information Processing Systems*.

Wei Chu, Zoubin Ghahramani, Francesco Falciani, and David L Wild. 2005. Biomarker discovery in microarray gene expression data with Gaussian processes. *Bioinformatics*, 21(16):3385–93, August.

Trevor Cohn and Lucia Specia. 2013. Modelling Annotator Bias with Multi-task Gaussian Processes: An Application to Machine Translation Quality Estimation. In *Proceedings of ACL*.

James Hensman, Nicolò Fusi, and Neil D. Lawrence. 2013. Gaussian Processes for Big Data. In *Proceedings of UAI*.

Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian processes for machine learning*, volume 1. MIT Press Cambridge.

Kashif Shah, Trevor Cohn, and Lucia Specia. 2013. An Investigation on the Effectiveness of Features for Translation Quality Estimation. In *Proceedings of MT Summit XIV*.

Edward Snelson and Zoubin Ghahramani. 2006. Sparse Gaussian Processes using Pseudo-inputs. In *Proceedings of NIPS*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*.

Lucia Specia, Craig Saunders, Marco Turchi, Zhuoran Wang, and John Shawe-Taylor. 2009. Improving the confidence of machine translation quality estimates. In *Proceedings of MT Summit XII*.

Lucia Specia, Kashif Shah, José G. C. De Souza, and Trevor Cohn. 2013. QuEst - A translation quality estimation framework. In *Proceedings of ACL Demo Session*.

Lucia Specia. 2011. Exploiting objective annotations for measuring translation post-editing effort. In *Proceedings of EAMT*.