Finding Convincing Arguments using Scalable Bayesian Preference Learning

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Abstract

Argumentative texts vary greatly in quality, meaning that identifying the most convincing arguments on each side of a controversial topic could be highly beneficial for decision making. To address this problem, we present a scalable Gaussian process approach for preference learning and apply the method to predicting argument convincingness. While Bayesian methods have been shown to be effective when dealing with small and noisy datasets, methods such as Gaussian processes (GP) have not previously been applied to argument quality. A perceived drawback is a lack of scalability, which we address through a new inference method for GP preference learning using recent advances in stochastic variational inference. We show how our method can be applied to predict argument convincingness from crowdsourced data, outperforming state-of-the-art methods, particularly when the data is sparse or noisy. We demonstrate more effective active learning using our Bayesian approach, thereby reducing the amount of data required to identify convincing arguments for new users and domains. Our results also illustrate the value of combining linguistic features with word embeddings to improve performance.

1 Introduction

Argumentation is intended to persuade the reader of a particular point of view and is an important way for humans to reason about controversial topics (Mercier and Sperber, 2011). The amount of argumentative text on any given topic can, however, overwhelm a reader, particularly considering the scale of historical text archives and the prevalence of social media platforms with millions of authors. To gain an understanding of a topic it is therefore useful to identify high-quality, persuasive arguments from different sides of a debate. Whether an argument is persuasive or not is subjective (Lukin et al., 2017), hence analysing which arguments a particular person or group of people finds convincing can tell us about their opinions and influences.

Previous work (Habernal and Gurevych, 2016b) showed that it is possible to predict the convincingness of arguments taken from online discussion forums with reasonable accuracy, using models trained on one topic and transferred to another. Their experiments made use of pairwise preference labels indicating which argument in a pair the annotator thought was more convincing. As a means of eliciting convincingness, pairwise preferences have a number of advantages. Unlike ratings or scores, they do not require calibrating, even if multiple people provide the labels, e.g. to mitigate the fact that some annotators may avoid very high or very low ratings, or may be biased toward particular scores. Pairwise comparisons are also more fine-grained than categorical labels and can lead to more reliable results with less cognitive burden on human annotators(Kendall, 1948; Kingsley, 2006). Implicit preferences can also be elicited from user actions, such as selecting a document from a list given its summary to read in more detail(Joachims, 2002).

In practice, however, preference data may be noisy – particularly if obtain from crowds or implicit feedback – and we may be faced with very small amounts of data when we move to new domains, topics and users for whom we wish to predict convincingness. Small data can present a problem to methods such as deep neural networks(Srivastava et al., 2014). The approach used by (Habernal and Gurevych, 2016b) to handle unreliable crowdsourced data involved first determining consensus labels using MACE(Hovy et al., 2013) and then ranking using PageRank to obtain training data for regression. Such pipeline approaches can be prone to error propagation(Chen and Ng, 2016) and require multiple crowdsourced labels for each argument pair to avoid individual errors.

In contrast to previous work, we propose the use of preference learning techniques for argument convincingness to directly model the relationship between crowdsourced preferences and textual features, including word embeddings. We choose a Bayesian approach, since Bayesian methods have been shown to successfully handle the problem of small(for example, (Xiong et al., 2011; Titov and Klementiev, 2012)) and unreliable datasets (e.g. (Simpson et al., 2015)), and provide a good basis for active selection to reduce labelling costs(MacKay, 1992). Our method is based on the Gaussian process (GP) model of (Chu and Ghahramani, 2005), which assumes that preferences over items are described by a latent preference function. By providing a Bayesian treatment to this latent function, the method handles uncertainty in the function values due to noise and data sparsity in a principled manner. GP preference learning (GPPL) has not previously been applied to text problems with large numbers of features and the inference scheme proposed by (Chu and Ghahramani, 2005) was limited by a computational complexity of $\mathcal{O}(N^3)$, where N is the number of items. We address the problem of scalability by applying recent advances in stochastic variational inference (SVI) (Hoffman et al., 2013) to this model, and developing an efficient optimisation technique for key hyper-parameters. We then show how our method can be applied to argument convincingness with a large number of linguistic features and high-dimensional text embeddings. Our evaluation compares Bayesian preference learning to established SVM and neural network approaches for predicting convincing arguments, and show that our approach can outperform these alternatives particularly with small and noisy datasets.

The rest of the paper is structured as follows. First, we review related work in more detail: on argumentation; Bayesian methods for preference learning; and scalable approximate inference. We then explain the preference learning approach in detail and develop our SVI inference and hyper-parameter optimisation methods. The following section details a number of experiments: a comparison with the state-of-the art on predicting preference in online debates; noisy dataset; active learning; and feature relevance determination. Finally, we present some conclusions and avenues for future work.

2 Related Work

Recent work on argumentation by (Habernal and Gurevych, 2016b) has established datasets and methods for predicting which argument is most convincing. Our experiments make extensive use of this data to establish a different methodology. This work was also extended to evaluate the reasons why one argument is more convincing than another(Habernal and Gurevych, 2016a), however our paper focusses on prediction when reasons are not given. Investigations by (Lukin et al., 2017) demonstrated the effect of personality and prior stance of the audience on the persuasiveness of arguments, although their work does not extend to modelling this persuasiveness using preference learning. The sequence of arguments in a dialogue is another important factor in their ability to change the audience's opinions (Tan et al., 2016). This idea is used by (Rosenfeld and Kraus, 2016; Monteserin and Amandi, 2013), who address the problem of choosing the best argument in a dialogue between a human user and an agent. However, these works focus on applying reinforcement learning to predict the best argument to present in a sequence rather than learning user preferences for arguments with certain qualities.

The goal of preference learning is to predict a ranking over items in terms of preference, or to predict which single item x_i in a pair or small set would be chosen by the user. A preference for item x_i over x_j is written as $x_i \succ x_j$. Given a ranking over items, it is possible to determine the pairwise preferences, but pairwise labels can also be predicted using a generic classifier without the need to learn a total ordering. During training and prediction, pairs of items are transformed either by concatenating the feature vectors of two items as in (Habernal and Gurevych, 2016b), or computing the difference of the two feature vectors as in SVM-Rank(Joachims, 2002). The classifier is then trained as normal with preference labels treated as binary class labels.

However, the ranking of items is useful for producing ordered lists in response to a query – consider a sorted list of the most convincing arguments in favour of topic X. Another approach is to learn this ordering directly using Mallows models(Mallows, 1957), which define distributions over permutations of a list. Mallows models have been extended to provide a generative model(Qin et al., 2010) and to be trained from pairwise preferences rather than by observing rankings(Lu and Boutilier, 2011). A disadvantage of Mallows models is that inference is typically costly, since the number of possible permutations to be considered is $\mathcal{O}(N^2)$, where N is the number of items to be ranked. Modelling only the order of items means we are unable to quantify how closely rated items at similar ranks are to one another: how much better is the top ranked item from the second-rated?

To avoid the problems of classifier-based and permutation-based methods, another approach is to learn a set of underlying real-valued scores from pairwise labels. These scores can then be used to predict rankings, pairwise labels, or ratings for individual items. To do this, a model is required to map the real-valued scores to discrete pairwise labels. Two established approaches for this are based on the Bradley-Terry-Plackett-Luce model (Bradley and Terry, 1952; Luce, 1959; Plackett, 1975) and the Thurstone-Mosteller model(Thurstone, 1927; Mosteller, 2006). In more recent work, Bayesian extensions of the Bradley-Terry-Plackett-Luce model were proposed by (Guiver and Snelson, 2009; Volkovs and Zemel, 2014), while the Thurstone-Mosteller model was used by (Chu and Ghahramani, 2005). This latter piece of work assumes a Gaussian process (GP) prior over the scores, which enables us to predict scores for previously unseen items given their features using a Bayesian nonparametric approach. Gaussian processes have been well established as effective and versatile models that extrapolate from training data in a principled manner, taking into account model uncertainty (Rasmussen and Williams, 2006). Their nonparametric nature means that the function complexity can grow with the amount of data observed. These characteristics make them suitable for the task of modelling argument convincingness where data for new topics, domains and users is limited.

The method developed by (Chu and Ghahramani, 2005) used the Laplace approximation to perform approximate inference over the model. Unfortunately the memory and computational costs scale with $\mathcal{O}(N^3)$ due to matrix inversion. If this limitation is overcome, there is still a computational and memory cost during training of $\mathcal{O}(N^2)$ due to the number of pairs in the training dataset. Such problems are common when performing inference over Gaussian process models but have been addressed by (Hensman et al., 2013; Hensman et al., 2015) for regression and classification tasks using the stochastic variational inference (SVI) algorithm proposed by (Hoffman et al., 2013). SVI has, however, not previously been adapted for preference learning with GPs. The next section explains our preference learning method for argument convincingness.

3 Scalable Bayesian Preference Learning

Following (Chu and Ghahramani, 2005), we model the relationship between a latent preference function, f, and each observed pairwise label, $[v_k \succ u_k]$, where k is an index into a list of P pairs, as follows:

$$p(v_k \succ u_k | f(v_k), f(u_k), \delta_{v_k}, \delta_{u_k}) \\ = \begin{cases} 1 & \text{if } f(v_k) + \delta_{v_k} \geq f(u_k) + \delta_{u_k} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\delta_i \sim \mathcal{N}(0,1)$ is Gaussian-distributed noise. The noise term allows for variations in the observed preferences, which may occur if multiple people are asked to provide a label, if the annotator is uncertain and changes her mind, or if the preferences are derived from noisy implicit data such as clicks streams. The unobserved noise terms are integrated out to obtain:

$$p(v_k \succ u_k | f(v_k), f(u_k)) = \Phi\left(\frac{f(v_k) - f(u_k)}{\sqrt{2}}\right), \tag{2}$$

where Φ is the cumulative distribution function of a standard Gaussian distribution. We deviate from the definition in (Chu and Ghahramani, 2005) because we assume that δ_i has a variance $\sigma = 1$, and instead learn the scale of the function f. This formulation is equivalent since the relative scales of the noise δ_i and the preference function f are maintained.

The goal of inference is to learn the posterior distribution over the function values f_i for each item i. We assume a Gaussian process prior: $f \sim \mathcal{GP}(0, k_\theta/s)$, where the terms are explained as follows: k_θ is a kernel function with hyper-parameters θ, which effectively controls the correlation between values of f at different points in the feature space; $s \sim \mathcal{G}(a_0, b_0)$ is an inverse scale parameter that controls the level of noise in the function and has a gamma prior with shape a_0 and scale b_0.

Inference in (Chu and Ghahramani, 2005) was performed using gradient descent to optimise a Laplace approximation. However, this approach produced a maximum a-posteriori (MAP) approximation, which takes the most probable point values of parameters rather than integrating over their distributions, and thus does not give a fully Bayesian treatment of parameter uncertainty and has been shown to perform poorly for tasks such as classification (Nickisch and Rasmussen, 2008). Furthermore, the presented approach is not scalable as it requires $\mathcal{O}(n^3)$ computation and $\mathcal{O}(n^2)$ memory, where n is the number of observations. We address this problem by first adapting the variational inference method based on the extended Kalman filter (EKF) (Reece et al., 2011; Steinberg and Bonilla, 2014) to the preference likelihood given by Equation 2. We refer to a particular pair of items by the index k, and all pairwise labels for that specific pair as $\boldsymbol{y}_k$. To permit inference using the variational method, our model approximates a Beta distribtion over the observation likelihood using a Gaussian distribution: $p(v_k \succ u_k|\boldsymbol{y}_k) \approx \mathcal{N}(v_k \succ u_k|\Phi(\hat{f}_{v_k} - \hat{f}_{u_k}/\sqrt{2}), \nu_k)$. The variances v_k for all pairs form a diagonal matrix $\boldsymbol{Q}$ and are estimated by moment matching with the variance of a beta distribution $\mathbb{V}[\nu_k] \sim \mathcal{B}(1+\sum_{i=1}^{P_k} y_{k,i}, 1+\sum_{i=1}^{P_k} 1-y_{k,i})$, where P_k is the number of pairwise labels for the items v_k and u_k in pair k. Further details are given in (Reece et al., 2011). This approximation means that the posterior distribution over f for items in the training set is also Gaussian: $p(f(\boldsymbol{x})|\boldsymbol{y}) \approx \mathcal{N}(f(\boldsymbol{x})|\hat{\boldsymbol{f}}, \boldsymbol{C})$ where $\boldsymbol{x}$ is a matrix of input features for the training items.

We now use variational inference to iteratively optimise the approximate posterior parameters $\hat{\boldsymbol{f}}$ and $\boldsymbol{C}$ and the posterior distribution over the inverse scale s. The algorithm we use maximises a lower bound, $\mathcal{L}$, on the log marginal likelihood, $p(\boldsymbol{y}|\theta, a_0, b_0)$:

$$\begin{aligned}
\mathcal{L}(q) \approx & -\frac{1}{2}\{L\log 2\pi + \log|\boldsymbol{Q}| - \log|\boldsymbol{C}| + \log|\boldsymbol{K}| \\
& +(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu}) \\
& +(\boldsymbol{y} - \Phi(\hat{\boldsymbol{z}}))^T\boldsymbol{Q}^{-1}(\boldsymbol{y} - \Phi(\hat{\boldsymbol{z}}))\} \\
& + \Gamma(a) - \Gamma(a_0) + a_0(\log b_0) + (a_0 - a)\widehat{\ln s} \\
& + (b - b_0)\hat{s} - a\log b, \qquad (3)
\end{aligned}$$

where L is the number of observed preference labels, $\boldsymbol{y} = [[v_1 \succ u_1], ..., [v_L \succ u_L]]$ is a vector of binary labels indicating whether v_i was preferred to u_i, $\widehat{\ln s} = \mathbb{E}[\log s]$ and $\hat{s}$ are expected values given the approximate posterior distribution over s, and we compute $\hat{\boldsymbol{z}}$ using:

$$\hat{\boldsymbol{z}} = \left\{ \frac{\hat{f}_{v_k} - \hat{f}_{u_k}}{\sqrt{2}} \forall k = 1, ..., P \right\}. \qquad (4)$$

To provide a scalable algorithm, we adapt stochastic variational inference (SVI) (Hensman et al., 2013; Hensman et al., 2015) from classification tasks to preference learning. For SVI we assume M *inducing points* with features $\boldsymbol{x}_m$. By choosing a value of $M << N$ that is much smaller than our dataset, we can limit the computational and memory requirements for performing approximate inference. The inducing points act as a substitute for the real feature vectors of the observed arguments. Therefore, to choose representative inducing points, we use a cheap clustering algorithm to identify cluster centres that can be used as inducing points. In our experiments we used K-means.

Given the inducing points, SVI further limits computational costs by using an iterative update algorithm that performs calculations involving only $P_m << P$ pairs at each iteration. As with M, the value of P_m can be chosen by the developer to fit their hardware requirements. Smaller values will consider less data at each iteration and therefore may require a larger number of iterations to converge. However, convergence is hard to predict and depends on the properties of the dataset used. The algorithm begins by randomly initialising estimates of the mean at the inducing points, $\hat{\boldsymbol{f}}_m$, the covariance of the inducing points, $\boldsymbol{S}$, the inverse function scale expectations $\hat{s}$ and $\hat{\ln s}$, and the Jacobian of the pairwise label probabilities, $\boldsymbol{G}$. The latter is required to enable a first order Taylor series approximation, which makes the iterative updates tractable. Then, at each iteration n of the SVI algorithm, these values are updated using the following equations, which are adapted from the derivations by (Hensman et al., 2013; Hensman et al., 2015):

$$\boldsymbol{S}_n^{-1} = (1-\rho_n)\boldsymbol{S}_{n-1}^{-1} + \rho_n \left(w_n \boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{nm}^T\boldsymbol{G}^T\boldsymbol{Q}^{-1}\boldsymbol{G}\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-T} + \hat{s}\boldsymbol{K}_{mm}^{-1}\right) \tag{5}$$

$$\hat{\boldsymbol{f}}_{m,n} = \boldsymbol{S}_n \left((1-\rho_n)\boldsymbol{S}_{n-1}^{-1}\hat{\boldsymbol{f}_{m,n-1}} + w_n\rho_n\boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{nm}^T\boldsymbol{G}^T Q^-1 \left(\frac{1+\sum_{l=1}^{P_k} y_{k,l}}{P_k} - \Phi(\hat{\boldsymbol{z}}_n) - \boldsymbol{G}\hat{\boldsymbol{f}}\right)\right) \tag{6}$$

$$\hat{s} = a/b, \qquad \hat{\ln} s = \Psi(a) - \log(b) \tag{7}$$

$$\boldsymbol{G} = \frac{1}{2\pi}\exp\left(-\frac{1}{2}\hat{\boldsymbol{z}}_n^2\right) \tag{8}$$

where $\boldsymbol{K}_{nm}$ is the covariance between the subsample of observations at iteration n and the inducing points; $\hat{\boldsymbol{z}}_n$ is the preference label likelihood for nth subsample of observations given by Equation 4; s has a gamma distribution with shape $a = a_0 + \frac{N}{2}$ and inverse scale $b = b_0 + \frac{1}{2}\mathrm{Tr}\left(\boldsymbol{K}_j^{-1}\left(\boldsymbol{\Sigma}_j + \hat{\boldsymbol{f}}_j\hat{\boldsymbol{f}}_j^T - 2\boldsymbol{\mu}_{j,i}\hat{\boldsymbol{f}}_j^T - \boldsymbol{\mu}_{j,i}\boldsymbol{\mu}_{j,i}^T\right)\right)$, Ψ is the digamma function; the term $\rho_i = (n + \mathrm{delay})^{-\mathrm{forgetting}_\mathrm{r}\mathrm{ate}}$ controls the combination of estimates between iterations; finally, the weight $w_n = \frac{N}{N_{\mathrm{subsample}}}$ weights the update according to the size of the subsample of observations. Equations 5 to 8 are repeated until convergence. Given the converged estimates, we can make predictions for test arguments with feature vectors $\boldsymbol{x}_*$ according to:

$$\hat{\boldsymbol{f}}_* = \boldsymbol{K}_{*m}\boldsymbol{K}_{mm}^{-1}\hat{\boldsymbol{f}}_m \tag{9}$$

$$\mathbb{V}[\boldsymbol{f}_*] = \boldsymbol{K}_{**}/\hat{s} + (\boldsymbol{K}_{*m}\boldsymbol{K}_{mm}^{-1}\boldsymbol{S} - \boldsymbol{K}_{*m}\boldsymbol{K}_{mm}^{-1})\boldsymbol{K}_{mm}^{-T}\boldsymbol{K}_{*m}^T, \tag{10}$$

where $\boldsymbol{K}_{*m}$ is the covariance between the test feature vectors and the inducing points, $\boldsymbol{K}_{mm}$ is the covariance between test feature vectors, and $\hat{\boldsymbol{f}}_*$ and $\mathbb{V}[\boldsymbol{f}_*]$ are the posterior mean and variance of preference function values for the test arguments.

3.1 Kernel Length-scale Optimsation

Typically, the kernel hyper-parameters θ contains a length-scale for each feature, l, which controls the smoothness of the function across the feature space. The kernel is a function of distance between two feature vectors and the length-scale, e.g. $k_\theta(\boldsymbol{x}, \boldsymbol{x}') = k(|\boldsymbol{x} - \boldsymbol{x}'|/l)$. This is therefore an important hyper-parameter and determines whether the model performs well or not. The median heuristic is one choice that has been shown to work well in practice (Gretton et al., 2012): $l_{MH} = \frac{1}{D}\mathrm{median}(\{|\boldsymbol{x}_i - \boldsymbol{x}_j'|\forall i = 1,..,N\forall j = 1,...,N\})$. The length-scales can also be optimised by choosing the values that maximise the lower bound on the log marginal likelihood, $\mathcal{L}$, defined in Equation 3. This process is known as maximum likelihood II (Rasmussen and Williams, 2006). Features that have very large length-scales can be considered irrelevant, since the value of k_θ is almost independent of that feature. The process of optimising length-scales is therefore referred to as automatic relevance determination (ARD). Removing irrelevant features could improve performance, since it reduces the dimensionality of the space of the preference function. A problem when using text data is that large vocabulary sizes and additional linguistic features lead to a large number of dimensions, D. The standard maximum likelihood II optimisation requires $\mathcal{O}(D)$ operations to tune each length-scale. This cost can be reduced by simultaneously optimising all length-scales using a gradient-based method such as L-BFGS-B (Zhu et al., 1997). To enable such an approach, we compute the gradients of $\mathcal{L}(q)$ with respect to the length-scale of each feature dimension, l_d. These gradients are given by:

$$\nabla_{l_d}\mathcal{L}(q) = \frac{1}{2}\hat{s}\hat{\boldsymbol{f}}_m^T \boldsymbol{K}_{mm}^{-T}\frac{\partial \boldsymbol{K}_{mm}}{\partial l_d}\boldsymbol{K}_{mm}^{-1}\hat{\boldsymbol{f}}_m - \frac{1}{2}\mathrm{tr}\left(\left(\hat{s}\boldsymbol{K}_{mm}^{-1}\boldsymbol{S}\right)^T\left(\boldsymbol{S}^{-1} - \boldsymbol{K}_{mm}^{-1}/\hat{s}\right)\frac{\partial \boldsymbol{K}_{mm}}{\partial l_d}\right) . \tag{11}$$

For the Matèrn $\frac{3}{2}$ kernel, which we use in our experiments due to its general properties of smoothness(Rasmussen and Williams, 2006), we compute:

$$\frac{\partial \boldsymbol{K}}{\partial l_d} = \prod_{d'=1,d'\neq d}^{D} K_d \frac{\partial K_{l_d}}{\partial l_d} \tag{12}$$

$$\frac{\partial K_{l_d}}{\partial l_d} = \frac{3|\boldsymbol{x}_d - \boldsymbol{x}'_d|^2}{l_d^3}\exp\left(-\frac{\sqrt{3}|\boldsymbol{x}_d - \boldsymbol{x}'_d|}{l_d}\right), \tag{13}$$

where $|\boldsymbol{x}_d - \boldsymbol{x}'_d|$ is the distance between input points. Here, for reasons of scalability, we continue to work in terms of inducing points.

UKPConvArgStrict	Combine crowdsourced labels with MACE and take $\geq$ 95% most confident labels; Discard arguments marked as equally convincing; Discard conflicting preferences.
UKPConvArgAll	Combine crowdsourced labels with MACE and take $\geq$ 95% most confident labels;
UKPConvArgRank	Combine crowdsourced labels with MACE and take $\geq$ 95% most confident labels; PageRank used to produce ranking for each topic.
UKPConvArgCrowdSample	One original crowdsourced label per pair; PageRank used to produce ranking for each topic.

Table 1: Summary of the processing steps used to produced the gold-standard data for each fold in each different dataset.

4 Experiments

To evaluate our approach, we consider a number of different scenarios in which we test both classification performance, i.e. predicting binary pairwise labels, and ranking performance, i.e. predicting the order of preference of arguments. We begin with a synthetic data experiment to illustrate how key methods work, then evaluate the methods on real crowdsourced data for argumentation. provided by (Habernal and Gurevych, 2016b), which was obtained using crowdsourcing. The crowdsourced datasets contain pairwise preference labels for arguments taken from online discussion forums. Each pairwise label indicates "which argument is more convincing", or may express no preference. We use four variants of this data, each of which involves different pre-processing steps. All datasets contain 32 folds, which correspond to 16 controversial topics, and two stances for each topic. The differences between the datasets are shown in Table 1.

A major aim of our experiments is to compare our Bayesian preference learning method, which we refer to here as *GPPL*, against the *SVM* and *BLSTM* methods used in (Habernal and Gurevych, 2016b). For classifications, SVM and BLSTM concatenates the feature vectors of each pair of arguments. For ranking, SVM and BLSTM are used to perform regression and are trained on the output of the PageRank model run over items in the training folds. As well as comparing classification and ranking performance, we also investigate how each method resolves conflicting preference pairs in crowdsourced data; which types of input features are useful for modelling convincingness; how well each method estimates uncertainty and how well it performs with active learning to address cold-start problems; and how well does each method cope with data sparsity, e.g. in a cold-start situation, and noisy data.

4.1 Experiment 1: Toy Data

Our two tasks are to *score* arguments in terms of convincingness and to *classify* the preference label for a pair of arguments, i.e. predict which of the arguments will be preferred. We use simulated data to show how Gaussian process preference learning differs from the established approaches for each task, namely SVM for the classification task and PageRank for the scoring task. Our simulation consists of four scenarios; in each scenario, we assume a set of pairwise preference labels for arguments labelled arg0 to arg4. The pairwise labels are depicted as convincingness graphs in Figure 1a. Arrows indicate the preferred arguments, e.g. the first plot shows that arg3 is more convincing than arg4. Each scenario is repeated 25 times and in each run we select arguments at random from one fold of the UKPConvArgStrict, then associate these arguments with the labels arg0 to arg4. For each argument, we obtain a feature vector by computing mean Glove word embeddings as in (Habernal and Gurevych, 2016b). We trained PageRank, GPPL and the SVM classifier on the preference pairs shown in each graph. The PageRank scores and GPPL latent preference function means for each argument are shown in Figure 1b. The first scenario, ”no cycle” shows a simple dataset where arg0 is preferred to both arg1 and arg2, which is reflected in both the PageRank and GPPL scores in Figure 1b. However, arg3 and arg4 are not connected to the rest of the graph and receive different scores with PageRank and GPPL. The classifications for pairs of arguments produced by GPPL and SVM are shown in Figures 1c and 1d. GPPL provides probabilistic classifications that give less confident estimates for many of the pairs that were not yet observed, e.g. $2 \succ 4$.

The second scenario, ”single cycle”, shows how each method handles a cycle in the preference graph. Both PageRank and GPPL produce even values for the arguments in the cycle (arg0, arg1 and arg2). PageRank assigns lower scores to both arguments that are not in the cycle (arg3 and arg4), while GPPL gives a higher score only to the preferred argument, arg3. The SVM predictions for ”single cycle” predict that arg0 and arg1 are preferred over arg3, although arg0 and arg1 are in a cycle and it is unclear why arg0 and arg1 would be preferred. GPPL in contrast gives a weak prediction that arg3 is preferred.

The third scenario, ”double cycle” produces very different results with PageRank and GPPL. Here, the argument graph shows two paths from arg2 to arg0 via arg1 or arg3, and one conflicting preference arg2 $\succ$ arg0. GPPL scores the arguments as if the single conflicting preference, arg2 $\succ$ arg0, was not present, likely giving more weight to two parallel paths from arg2 to arg0. In contrast, PageRank gives high scores to both arg0 and arg2. The classifications by GPPL and SVM are similar, but GPPL produces more uncertain predictions than in the first scenario, likely due to the conflicting edge.

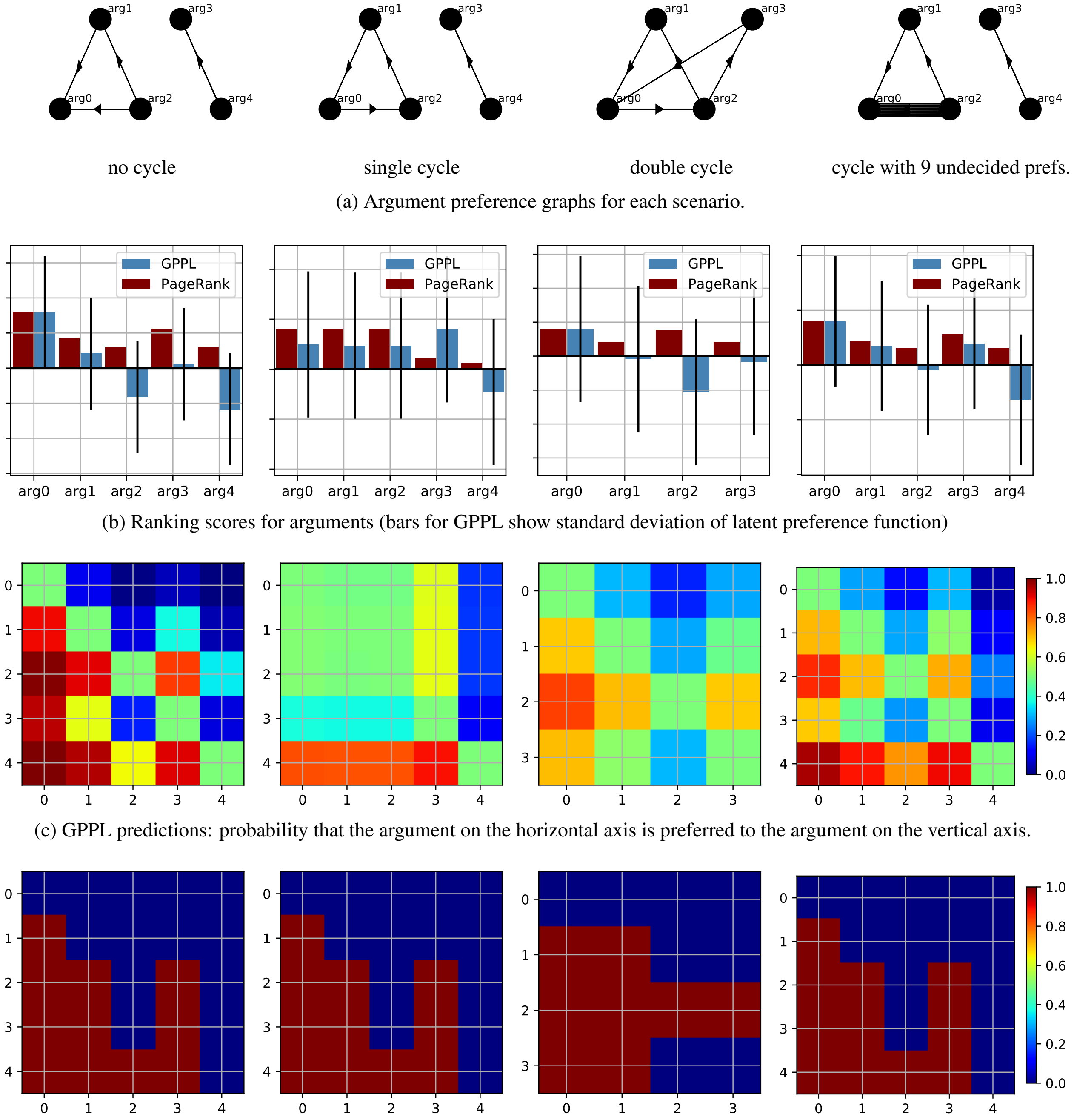
Finally, ”cycle with 9 undecided prefs” shows an exaggerated scenario in which we have added nine ”undecided” preference labels to the ”no cycle” scenario. The undirected edges indicate that neither argument was preferred and simulate the case where multiple annotators were asked to label a pair and did not all agree. This does not affect the PageRank scores, but reduces the difference in GPPL scores between arg0 and other arguments, since the preference arg2 $\succ$ arg0 is effectively given less weight due to the undecided labels. This is reflected in the predicted classifications from GPPL, which are less confident than in the ”no cycle” scenario. The SVM cannot be trained using the uncertain labels and therefore does not adapt to the undecided labels.

In conclusion, GPPL appears to resolve conflicts in the preference graphs in a more intuitive manner than PageRank, which was designed for ranking web pages by importance and therefore may be less suitable for ranking by preference. In contrast to SVM, it is able to account for undecided labels to soften the underlying convincingness function.

4.2 Experiment 2: Clean Data

We first compare GPPL to the SVM and BLSTM methods for classification on the UKPConvArgStrict dataset and ranking on the UKPConvArgRank dataset. Both datasets were cleaned to remove disagreements between annotators. We compared GPPL with different sets of input features with the length-scale set using the median heuristic: first, we use 32000 linguistic features (labelled as *ling*) that we also use for the SVM (as in (Habernal and Gurevych, 2016b)), second, the Glove word embeddings with 300 dimensions that we also use for the BLSTM (also as in (Habernal and Gurevych, 2016b), labelled *Glove*), and finally, the combination of both linguistic features and embeddings (labelled *ling + Glove*). To create embeddings for an argument, we take the mean of the individual word embeddings for the tokens in the argument.

We also compare multiplicative and additive combinations for the kernel functions and test three different settings for the hyper-parameters a_0 and b_0, which control the noise variance. While it may be



(a) Argument preference graphs for each scenario.

(b) Ranking scores for arguments (bars for GPPL show standard deviation of latent preference function)

(c) GPPL predictions: probability that the argument on the horizontal axis is preferred to the argument on the vertical axis.

(d) SVM predictions: probability that the argument on the horizontal axis is preferred to the argument on the vertical axis.

Figure 1: Preference graphs and predictions for simulated arguments in different scenarios. The plots in each column correspond to a single scenario.

possible to further optimise a_0 and b_0, we found weak priors favouring a moderate level of noise to be most effective and therefore continue to use $a_0 = 2$, $b_0 = 200$ in the remaining experiments. We also tested $a_0 = 1$, $b_0 = 1$, which favours lower noise variance, but this resulted in a cross entropy error (CEE) of 0.69 because the predicted classifications were very close to 0.5, compared to CEE of 0.47 when using $a_0 = 2$, $b_0 = 200$. With stronger priors $a_0 = 2e3$, $b_0 = 2e5$, we observed a drop in accuracy of 7% on UKPConvArgStrict so concluded that $a_0 = 2$, $b_0 = 200$ is a reasonable setting to proceed without further optimisation.

The results are shown in Table 2. When using *ling* features, GPPL produces similar accuracy and improves the area under the ROC curve (AUC) by 2%, and cross entropy error by 0.01. Much larger improvements can be seen in the ranking metrics. When GPPL is run with mean Glove embeddings, it performs worse than BLSTM for classification but improves the ranking metrics. Using a combination of features, GPPL performs substantially better than the alternative methods for both classification and ranking, suggesting that embeddings and linguistic features contain complementary information.

We also apply the MLII optimisation method to GPPL with *ling+Glove* features. The results are labelled as ”OptGPPL” in Table 2, and show that optimising the length-scale using Bayesian model selection can further improve performance above the median heuristic. However, the cost of these improvements is that each fold required around 2 hours to compute instead of approximately 10 minutes on the same machine (an Intel i7 quad core desktop) using the median heuristic. The accuracy for each fold with and without length-scale optimisation is shown in Table 3, and shows that optimisation does not always improve performance. Since the optimisation step is performed using only the training folds and the test is performed on a different topic, there is a possibility of overfitting: features that are important in the test fold may not appear relevant in the training folds. Optimisation may therefore be more effective if it could be executed in a semi-supervised manner by including unlabelled data from the target topic.

We hypothesise that GPPL benefits from a Bayesian approach that integrates learning from discrete preference labels with regression over a latent preference function. To test this, we compare GPPL against a two stage method: first, we use use the GPPL preference likelihood method without any item features to infer convincingness scores for each argument from the pairwise labels; second, we perform SVM regression trained on the inferred scores with *ling+Glove* features. Hence we investigate whether the benefits of GPPL are entirely due its different preference likelihood in contrast to training the SVM classifier directly or using PageRank to estimate scores. The results are shown in Table 2 as *PL+SVR*, and show that while this approach does not reach the same performance as GPPL. This may be due to the benefits of integrating a GP to map the features to the latent function, rather than using a separate SVM regressor.

For the pairwise classification task, we also compare GPPL against a Gaussian process classifier (*GPC*) to investigate whether other GP-based approaches produce comparable performance. As shown in Table 2, GPC produces the best results on the classification task, although it cannot be used to rank the arguments. While the classification approach involves learning over twice as many features – the features of the first and second items in each pair are concatenated – the GPC may perform better on this dataset because it is trained directly on the classification task, rather than through a preference learning likelihood.

4.3 Experiment 3: Conflicts and Noisy Crowdsourced Data

In this experiment, we first introduced conflicts into the classification task by comparing methods on the UKPConvArgAll dataset, then additionally introduce noise to both the classification and the regression tasks by comparing on the UKPConvArgCrowdSample dataset. Our goal was to investigate whether a Bayesian approach is better able to handle noise and conflicts.

The results are shown in Table 4, showing that all methods produce lower performance on this dataset containing conflicting preferences. GPPL and GPC produce the best results, but GPC no longer has a clear advantage over GPPL. It is possible that GPC and SVM have the largest changes in accuracy compared to the UKPConvArgStrict results because

UKPConvArgStrict										
	SVM	BLSTM	GPPL*, medi.			GPPL*, opt	GPPL+, medi.	GPPL+, opt	PL +SVR	GPC
	ling	Glove	ling	Glove	ling+ Glove					
Acc.:	0.78	0.76	0.78	0.71	0.79	0.80	0.78	0.78	0.78	0.81
AUC:	0.83	0.84	0.85	0.77	0.87	0.87	0.86	0.86	0.85	0.89
CEE:	0.52	0.64	0.51	1.12	0.47	0.51	0.69	0.69	0.51	0.43
UKPConvArgRank										
Pears.:	0.36	0.32	0.38	0.33	0.45	0.44	0.40	0.40	0.39	-
Spear.:	0.47	0.37	0.62	0.44	0.65	0.67	0.64	0.64	0.63	-
Kend.:	0.34	0.27	0.47	0.31	0.49	0.50	0.49	0.49	0.47	-

Table 2: Performance comparison on clean datasets.

these classification-based methods have no mechanism to resolve conflicts in the preference graph. The performance of the BLSTM classifier also decreases by a smaller amount, but was already poorer than the other methods on UKPConvArgStrict so it is hard to compare this change directly. PL+SVR is again slightly poorer than GPPL and GPC.

When noise is introduced in the UKPConvArgCrowdSample dataset, most results drop further. GPPL now outperforms the other methods in all metrics except Spearman's ρ, where PL+SVR performs slightly better. The accuracy of GPC and SVM decreases, as a result of the noise that was introduced, while for other methods it remains the same as for UKPConvArgAll. Metrics for ranking on UKPConvArgCrowdSample show that while GPPL and PL+SVR continue to perform well, the results for BLSTM and particularly for SVM are much poorer than with UKPConvArgRank.

4.4 Experiment 4: Active Learning

We hypothesised that a Bayesian approach would deal better with sparse data and provide more meaningful confidence estimates. To test this hypothesis, we simulated an active learning scenario, in which we simulate an agent that iteratively learns a model for each fold. Initially, $N_{inc} = 2$ pairs were chosen at random from the training set, then used to train the classifier. The agent then performs *uncertainty sampling* to select the $N_{inc} = 2$ pairs with the least confident classifications. The labels for these pairs are then taken from the training set and used to re-train the model. The result is plotted in Figure 2, showing that GPPL is able to reach accuracies above 65% with only 50 labels, while SVM and BLSTM do not reach the same performance given 200 labels. The accuracy of GPPL also increases by approximately 8% given 200 labels, while SVM increases approximately 6% and BLSTM only 2%. This suggest that GPPL may be a more suitable model to be used with uncertainty sampling in situations where obtaining labelled data is expensive.

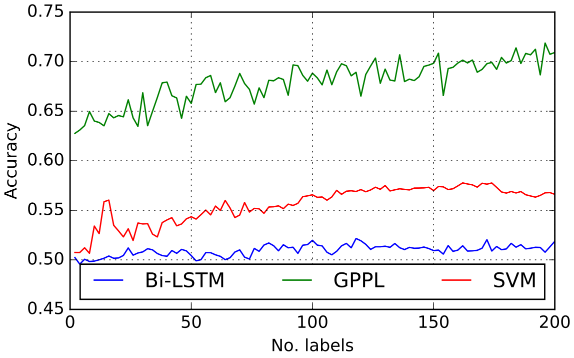


Figure 2: Active learning simulation for the three methods showing the mean accuracy of preference pair classifications over 32 runs.

4.5 Experiment 5: Embeddings

In our previous experiments, we found that including mean Glove word embeddings boosted performance above only using linguistic features. However, there are several alternative methods to mean word embeddings for representing longer pieces of text, notably skip-thoughts(**?**) and Siamese-CBOW(**?**). We compare mean Glove embeddings with skip-thoughts embeddings trained on the ...

UKPConvArgStrict									
	Median heuristic						Optimised		
	Glove	Skip-thoughts	SCBOW	ling+ Glove	ling+ Skip-th.	ling+ SCBOW	ling+ Glove	ling+ Skip-th.	ling+ SCBOW
Acc.:	0.71	0.67	0.69	0.79	0.74	0.77	0.80	0.78	0.78
AUC:	0.77	0.72	0.75	0.87	0.81	0.85	0.87	0.85	0.85
CEE:	1.12	1.11	1.22	0.47	0.80	0.52	0.51	0.51	0.50
UKPConvArgRank									
Pears.:	0.33	0.30	0.29	0.45	0.34	0.39	0.44	0.34	0.40
Spear.:	0.44	0.49	0.40	0.65	0.59	0.63	0.67	0.52	0.63
Kend.:	0.31	0.36	0.28	0.49	0.43	0.47	0.50	0.37	0.47

Table 5: Comparison between different types of embeddings with GPPL

corpus(**?**) and Siamese-CBOW embeddings trained on the ... corpus(**?**). The results are shown in Table 5 showing that the best performance was obtained using mean Glove embeddings. Further work may be required to assess whether skip-thoughts and Siamese-CBOW can be improved if trained on different corpora, or whether the mean word embeddings are simply more informative for predicting convincingness on our chosen dataset.

4.6 Experiment 6: Informative Features

Finally, we show how the length-scales learned by optimising GPPL can be used to identify informative sets of features. Since a larger length-scale causes greater smoothing, a very large length-scale implies that the value of that feature is irrelevant when predicting the function. In contrast, small length-scales indicate more informative features, since their precise value affects the latent preference function. Figure 3 shows the distribution of optimised length-scales on one fold of UKPConvArgStrict. The values shown are ratios of the optimised value to the median heuristic. Due to the computation time required, our optimisation procedure was limited to 25 function evaluations. The large number of values close to 1 may be due to the L-BFGS-B algorithm not being able to optimise all features in the available time, but it also suggests that other features with larger gradients were prioritised for optimisation, and hence that the median heuristic was a reasonable estimate for these features. The length-scales for many dimensions of the mean word embeddings were increased, giving ratios close to $4x$ the median heuristic, suggesting that these dimensions may be only very weakly informative. Table 6 shows the largest and smallest ratios for embeddings and linguistic features. The unigram ”safety” has a very high length-scale, suggesting it is not informative and may be discarded. It is possible that continuing the optimisation procedure for a larger number of steps would identify large length-scales for other features that may be discarded. However, caution is required to avoid overfitting to the training set during optimisation(**?**).

5 Conclusions and Future Work

We presented a novel, scalable approach to predicting argument convincingness using Bayesian preference learning, and demonstrated how our method can outperform SVM and neural network methods with sparse and noisy training data. Several related NLP and argumentation problems could benefit from a similar methodology, particularly in an interactive setting where large amounts of clean training data are unavailable. An example is the argument reasoning comprehension task(Ivan Habernal, 2017), where annotators select preferred components to complete an argument. Future work will therefore evaluate this preference learning approach on other NLP tasks where training data for standard classification and regression is unavailable.

The GP approach could also be used to combine information of different types: classifications and absolute scores as well as pairwise labels. The idea of combining scores and pairwise labels using an

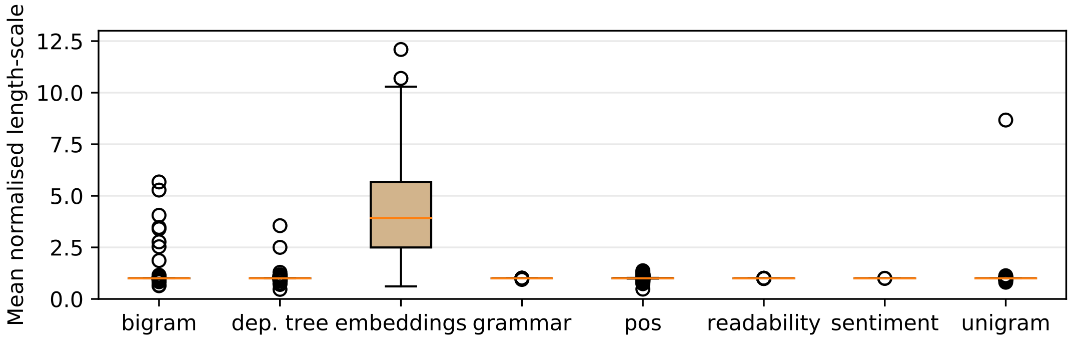


Figure 3: Distribution of length-scales for each type of feature after optimisation. Values are relative to the median heuristic value before optimisation, optimised on fold ”should physical education be mandatory in schools – no”, where optimisation increased accuracy from 75% to 80%.

active learning approach for image quality assessment with crowds was demonstrated by (Ye and Doermann, 2013), albeit using the Laplace approximation for the Gaussian process. In future we plan to investigate the effectiveness of this idea using our scalable Gaussian process approach for interactive learning settings with implicit feedback.

Acknowledgments

Topic	Stance	M.H.	Opt.
Ban plastic water bottles?	no	.81	.81
	yes	.83	.88
Christianity or atheism?	atheism	.83	.81
	Christianity	.78	.77
Evolution vs creation	creation	.87	.89
	evolution	.73	.72
Firefox vs Internet Explorer	IE	.86	.84
	firefox	.87	.86
Gay marriage – right or wrong?	right	.78	.76
	wrong	.86	.90
Should parents use spanking?	no	.85	.85
	yes	.73	.68
If your spouse committed murder...	no	.67	.69
	yes	.76	.72
India has the potential to lead the world	no	.82	.78
	yes	.84	.82
Lousy father better than fatherless?	no	.79	.76
	yes	.70	.65
Is porn wrong?	no	.80	.81
	yes	.89	.89
School uniform – good or bad idea	bad	.85	.83
	good	.74	.80
Pro choice vs pro life	pro choice	.69	.71
	pro life	.84	.80
Should physical edu. be mandatory?	yes	.83	.87
	no	.72	.75
TV is better than books	no	.81	.81
	yes	.82	.87
Personal pursuit or common good?	common	.78	.71
	personal	.66	.67
Farquhar the founder of Singapore?	no	.81	.80
	yes	.83	.66
Average		.79	.80

Table 3: Breakdown of accuracy by fold (topic and stance) for GPPL with different methods of choosing the length-scale (M.H. = median heuristic, Opt. = optimised).

UKPConvArgAll					
	SVM	B-LSTM	GPPL	PL+ SVR	GPC
	ling	Glove	ling+ Glove	ling+ Glove	ling+ Glove
Acc:	0.71	0.73	0.77	0.75	0.76
AUC:	0.81	0.81	0.84	0.82	0.86
CEE:	0.56	0.53	0.49	0.52	0.50
UKPConvArgCrowdSample					
Acc:	0.70	0.73	0.77	0.75	0.73
AUC:	0.81	0.80	0.84	0.82	0.86
CEE:	0.58	0.54	0.50	0.55	0.53
Pears.:	0.06	0.26	0.35	0.31	-
Spear.:	0.04	0.20	0.54	0.55	-
Kend.:	0.04	0.13	0.40	0.40	-

Table 4: Performance comparison on datasets containing conflicts and noise.

Feature	Ratio
ProductionRule-S->ADVP,NP,VP,.,	0.466
Pos-ngram-PP-O-CARD	0.477
Unigram-“safer”,	0.640
Bigram-“?”-“look”	5.672
Unigram-“safest”	8.673
Unigram-“safety”	271.190
Embedding-dimension-19	0.610
Embedding-dimension-241	12.093

Table 6: Ratios of optimised to median heuristic length-scales: largest and smallest ratios for linguistic features and word embeddings.

References

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Chen Chen and Vincent Ng. 2016. Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages. In *AAAI*, pages 2913–2920.

Wei Chu and Zoubin Ghahramani. 2005. Preference learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 137–144. ACM.

Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur. 2012. Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems*, pages 1205–1213.

John Guiver and Edward Snelson. 2009. Bayesian inference for plackett-luce ranking models. In *proceedings of the 26th annual international conference on machine learning*, pages 377–384. ACM.

Ivan Habernal and Iryna Gurevych. 2016a. What makes a convincing argument? empirical analysis and detecting attributes of convincingness in web argumentation. In *EMNLP*, pages 1214–1223.

Ivan Habernal and Iryna Gurevych. 2016b. Which argument is more convincing? Analyzing and predicting convincingness of Web arguments using bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1589–1599, Berlin, Germany. Association for Computational Linguistics.

James Hensman, Nicolò Fusi, and Neil D Lawrence. 2013. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 282–290. AUAI Press.

James Hensman, Alexander G. de G. Matthews, and Zoubin Ghahramani. 2015. Scalable Variational Gaussian Process Classification. In *AISTATS*.

Matthew D Hoffman, David M Blei, Chong Wang, and John William Paisley. 2013. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard H Hovy. 2013. Learning whom to trust with mace. In *HLT-NAACL*, pages 1120–1130.

Iryna Gurevych Benno Stein Ivan Habernal, Henning Wachsmuth. 2017. The argument reasoning comprehension task. *arXiv preprint arXiv:1708.01425*.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM.

Maurice George Kendall. 1948. Rank correlation methods.

David C Kingsley. 2006. Preference uncertainty, preference refinement and paired comparison choice experiments. *Dept. of Economics. University of Colorado*.

Tyler Lu and Craig Boutilier. 2011. Learning mallows models with pairwise preferences. In *Proceedings of the 28th international conference on machine learning (icml-11)*, pages 145–152.

R Duncan Luce. 1959. On the possible psychophysical laws. *Psychological review*, 66(2):81.

Stephanie Lukin, Pranav Anand, Marilyn Walker, and Steve Whittaker. 2017. Argument strength is in the eye of the beholder: Audience effects in persuasion. In *15th European Chapter of the Association for Computational Linguistics (EACL)*. Association for Computational Linguistics.

David JC MacKay. 1992. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604.

Colin L Mallows. 1957. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130.

Hugo Mercier and Dan Sperber. 2011. Why do humans reason? arguments for an argumentative theory. *Behavioral and brain sciences*, 34(2):57–74.

Ariel Monteserin and Analía Amandi. 2013. A reinforcement learning approach to improve the argument selection effectiveness in argumentation-based negotiation. *Expert Systems with Applications*, 40(6):2182–2188.

Frederick Mosteller. 2006. Remarks on the method of paired comparisons: I. the least squares solution assuming equal standard deviations and equal correlations. In *Selected Papers of Frederick Mosteller*, pages 157–162. Springer.

Hannes Nickisch and Carl Edward Rasmussen. 2008. Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078.

Robin L Plackett. 1975. The analysis of permutations. *Applied Statistics*, pages 193–202.

Tao Qin, Xiubo Geng, and Tie-Yan Liu. 2010. A new probabilistic model for rank aggregation. In *Advances in neural information processing systems*, pages 1948–1956.

C. E Rasmussen and C. K. I. Williams. 2006. Gaussian processes for machine learning. *The MIT Press, Cambridge, MA, USA*, 38:715–719.

Steven Reece, Stephen Roberts, David Nicholson, and Chris Lloyd. 2011. Determining intent using hard/soft data and gaussian process classifiers. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pages 1–8. IEEE.

Ariel Rosenfeld and Sarit Kraus. 2016. Providing arguments in discussions on the basis of the prediction of human argumentative behavior. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 6(4):30.

Edwin D Simpson, Matteo Venanzi, Steven Reece, Pushmeet Kohli, John Guiver, Stephen J Roberts, and Nicholas R Jennings. 2015. Language understanding in the wild: Combining crowdsourcing and machine learning. In *Proceedings of the 24th International Conference on World Wide Web*, pages 992–1002. International World Wide Web Conferences Steering Committee.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Daniel M Steinberg and Edwin V Bonilla. 2014. Extended and unscented gaussian processes. In *Advances in Neural Information Processing Systems*, pages 1251–1259.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th International Conference on World Wide Web*, pages 613–624. International World Wide Web Conferences Steering Committee.

Louis L Thurstone. 1927. A law of comparative judgment. *Psychological review*, 34(4):273.

Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22. Association for Computational Linguistics.

Maksims Volkovs and Richard S. Zemel. 2014. New learning methods for supervised and unsupervised preference aggregation. *Journal of Machine Learning Research*, 15(1):1135–1176.

Hui Yuan Xiong, Yoseph Barash, and Brendan J Frey. 2011. Bayesian prediction of tissue-regulated splicing using rna sequence and cellular context. *Bioinformatics*, 27(18):2554–2562.

Peng Ye and David Doermann. 2013. Combining preference and absolute judgements in a crowd-sourced setting. In *Proc. of Intl. Conf. on Machine Learning*, pages 1–7.

Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560.

A Variational Inference

We derive the variational lower bound as follows:

$$\begin{aligned}\mathcal{L}(q) = &\sum_{i=1}^{L} \mathbb{E}_q\left[\log p\left(v_i \succ u_i | f(v_i), f(u_i)\right)\right] \\ &+ \mathbb{E}_q\left[\log \frac{p\left(\boldsymbol{f}|\boldsymbol{\mu}, \boldsymbol{K}/s\right)}{q\left(\boldsymbol{f}\right)}\right] + \mathbb{E}_q\left[\log \frac{p\left(s|a_0, b_0\right)}{q\left(s\right)}\right]\end{aligned} \tag{14}$$

Substituting the forms of the distributions with their variational parameters, we get:

$$\begin{aligned}\mathcal{L}(q) =&\mathbb{E}_q\Bigg[\sum_{i=1}^{L}[v_i \succ u_i] \log \Phi(z_i) \\ &+ [v_i \prec u_i]\left(1 - \log \Phi(z_i)\right)\Bigg] \\ &+ \log \mathcal{N}\left(\hat{\boldsymbol{f}}; \boldsymbol{\mu}, \boldsymbol{K}/\hat{s}\right) - \log \mathcal{N}\left(\hat{\boldsymbol{f}}; \hat{\boldsymbol{f}}, \boldsymbol{C}\right) \\ &+ \mathbb{E}_q\left[\log \mathcal{G}\left(s; a_0, b_0\right) - \log \mathcal{G}\left(s; a, b\right)\right]\end{aligned} \tag{15}$$

We now replace the likelihood with a Gaussian approximation:

$$\begin{aligned}\mathcal{L}(q) \approx\ &\mathbb{E}_q\left[\mathcal{N}(\boldsymbol{y}|\Phi(\boldsymbol{z}), \boldsymbol{Q})\right] \\ &+ \log \mathcal{N}\left(\boldsymbol{f}; \boldsymbol{\mu}, \boldsymbol{K}/\hat{s}\right) - \log \mathcal{N}\left(\boldsymbol{f}; \hat{\boldsymbol{f}}, \boldsymbol{C}\right) \\ &+ \mathbb{E}_q\left[\log \mathcal{G}\left(s; a_0, b_0\right) - \log \mathcal{G}\left(s; a, b\right)\right] \\ \approx &-\frac{1}{2}\left\{L \log 2\pi + \log|\boldsymbol{Q}| - \log|\boldsymbol{C}|\right. \\ &+ \log|\boldsymbol{K}/s| + (\hat{\boldsymbol{f}} - \boldsymbol{\mu})\hat{s}\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu}) \\ &\left.+\mathbb{E}_q\left[(\boldsymbol{y} - \Phi(\boldsymbol{z}))^T\boldsymbol{Q}^{-1}(\boldsymbol{y} - \Phi(\boldsymbol{z}))\right]\right\} \\ &- \Gamma(a_0) + a_0(\log b_0) + (a_0 - a)\mathbb{E}[\log s] \\ &+ \Gamma(a) + (b - b_0)\hat{s} - a \log b\end{aligned} \tag{16}$$

Finally, we use a Taylor-series linearisation to make the remaining expectation tractable:

$$\begin{aligned}\mathcal{L}(q) \approx &-\frac{1}{2}\left\{L \log 2\pi + \log|\boldsymbol{Q}| - \log|\boldsymbol{C}|\right. \\ &+ \log|\boldsymbol{K}/\hat{s}| + (\hat{\boldsymbol{f}} - \boldsymbol{\mu})\hat{s}\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu}) \\ &\left.+(\boldsymbol{y} - \Phi(\hat{\boldsymbol{z}}))^T\boldsymbol{Q}^{-1}(\boldsymbol{y} - \Phi(\hat{\boldsymbol{z}}))\right\} \\ &- \Gamma(a_0) + a_0(\log b_0) + (a_0 - a)\mathbb{E}[\log s] \\ &+ \Gamma(a) + (b - b_0)\hat{s} - a \log b,\end{aligned} \tag{17}$$

where $\Gamma()$ is the gamma function, $\mathbb{E}[\log s] = \Psi(a) - \log(b)$, and $\Psi()$ is the digamma function.

The gradient of $\mathcal{L}(q)$ with respect to the lengthscale, l_d, is as follows:

$$\begin{aligned}\nabla_{l_d}\mathcal{L}(q) = &-\frac{1}{2}\left\{\frac{\partial \log|\boldsymbol{K}/\hat{s}|}{\partial l_d} - \frac{\partial \log|\boldsymbol{C}|}{\partial l_d}\right. \\ &\left.-(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\hat{s}\frac{\partial K^{-1}}{\partial l_d}(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\right\} \\ = &-\frac{1}{2}\left\{\frac{\partial \log|\frac{1}{\hat{s}}\boldsymbol{K}\boldsymbol{C}^{-1}|}{\partial l_d}\right. \\ &\left.+\hat{s}(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\boldsymbol{K}^{-1}\frac{\partial \boldsymbol{K}}{\partial l_d}\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\right\}\end{aligned} \tag{18}$$

Using the fact that $\log|A| = \mathrm{tr}(\log A)$, $\boldsymbol{C} = \left[\boldsymbol{K}^{-1} - \boldsymbol{G}\boldsymbol{Q}^{-1}\boldsymbol{G}^T\right]^{-1}$, and $\boldsymbol{C} = \boldsymbol{C}^T$, we obtain:

$$\begin{aligned}&= -\frac{1}{2}\mathrm{tr}\left(\left(\hat{s}\boldsymbol{K}^{-1}\boldsymbol{C}\right)\boldsymbol{G}\boldsymbol{Q}^{-1}\boldsymbol{G}^T\frac{\partial \boldsymbol{K}}{\partial l_d}\right) \\ &+ \frac{1}{2}\hat{s}(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\boldsymbol{K}^{-1}\frac{\partial \boldsymbol{K}}{\partial l_d}\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu}) \\ &= -\frac{1}{2}\mathrm{tr}\left(\left(\hat{s}\boldsymbol{K}^{-1}\boldsymbol{C}\right)\left(\boldsymbol{C}^{-1} - \boldsymbol{K}^{-1}/\hat{s}\right)\frac{\partial \boldsymbol{K}}{\partial l_d}\right) \\ &+ \frac{1}{2}\hat{s}(\hat{\boldsymbol{f}} - \boldsymbol{\mu})\boldsymbol{K}^{-1}\frac{\partial \boldsymbol{K}}{\partial l_d}\boldsymbol{K}^{-1}(\hat{\boldsymbol{f}} - \boldsymbol{\mu}).\end{aligned} \tag{19}$$

Assuming a product over kernels for each feature, $\boldsymbol{K} = \prod_{d=1}^{D}\boldsymbol{K}_d$, we can compute the kernel gradient as follows for the Matérn $\frac{3}{2}$ kernel function:

$$\frac{\partial \boldsymbol{K}}{\partial l_d} = \prod_{d'=1,d'\neq d}^{D} K_d \frac{\partial K_{l_d}}{\partial l_d} \tag{20}$$

$$\frac{\partial K_{l_d}}{\partial l_d} = \frac{3|\boldsymbol{x}_d - \boldsymbol{x}'_d|^2}{l_d^3}\exp\left(-\frac{\sqrt{3}|\boldsymbol{x}_d - \boldsymbol{x}'_d|}{l_d}\right) \tag{21}$$

where $|\boldsymbol{x}_d - \boldsymbol{x}'_d|$ is the distance between input points.