Machine Learning
https://doi.org/10.1007/s10994-021-05952-5

Large scale multi-label learning using Gaussian processes

Aristeidis Panos[1] · **Petros Dellaportas**[1,2,3] · **Michalis K. Titsias**[4]

Received: 18 December 2019 / Revised: 3 November 2020 / Accepted: 1 February 2021

Abstract

We introduce a Gaussian process latent factor model for multi-label classification that can capture correlations among class labels by using a small set of latent Gaussian process functions. To address computational challenges, when the number of training instances is very large, we introduce several techniques based on variational sparse Gaussian process approximations and stochastic optimization. Specifically, we apply doubly stochastic variational inference that sub-samples data instances and classes which allows us to cope with Big Data. Furthermore, we show it is possible and beneficial to optimize over inducing points, using gradient-based methods, even in very high dimensional input spaces involving up to hundreds of thousands of dimensions. We demonstrate the usefulness of our approach on several real-world large-scale multi-label learning problems.

Keywords Multi-label learning · Gaussian process · Variational inference · Bayesian nonparametrics

1 Introduction

Multi-label classification is a supervised learning problem where data instances are associated with multiple classes (Tsoumakas and Katakis 2007; Read et al. 2011; Zhang and Zhou 2013; Gibaja and Ventura 2014, 2015). It can be viewed as a generalization to the more traditional multi-class classification problem, where each data point can belong only to a single class. Multi-label learning has attracted a lot of attention in the recent literature,

Editor: Eyke Hüllermeier.

✉ Aristeidis Panos
aristeidis.panos.15@ucl.ac.uk

Petros Dellaportas
p.dellaportas@ucl.ac.uk

Michalis K. Titsias
mtitsias@aueb.gr

[1] Department of Statistical Science, University College London, London, UK

[2] Department of Statistics, Athens University of Economics and Business, Athens, Greece

[3] The Alan Turing Institute, London, UK

[4] Department of Informatics, Athens University of Economics and Business, Athens, Greece

Published online: 14 April 2021

due to its numerous applications ranging from text and image classification to computational advertising and recommender systems (Gibaja and Ventura 2014, 2015; Prabhu and Varma 2014; Jain et al. 2016). Two main challenges in multi-label learning are: (i) the modelling challenge associated with introducing suitable models to capture the correlation across different labels, and (ii) the computational or scalability challenge associated with dealing with datasets having very large number of labels, training instances and input dimensions.

In this paper, we tackle the problem of multi-label learning using a probabilistic framework based on Gaussian processes (GPs) (Rasmussen and Williams 2005). From a GP perspective multi-label learning shares similarities with the standard approaches for multi-task or multi-output Gaussian regression suitable for real-valued output data (Teh et al. 2005; Alvarez et al. 2012; Bonilla et al. 2008; Moreno-Muñoz et al. 2018). The difference is that in multi-label learning the output data are binary, thus requiring Bernoulli or binary regression type of likelihoods. Based on this, we introduce a multi-label extension of the semi-parametric latent factor model (Teh et al. 2005) that allows to capture the correlation of multiple labels using a small set of shared latent GP functions. Our work bear many similarities with Dai et al. (2017) and it can be seen as a special case of the more general model presented in Moreno-Muñoz et al. (2018) where both continuous and discrete outputs are allowed and a variant number of latent GP functions can be deployed.

Furthermore, to address the computational challenges when training the model, we make use of sparse GP approximations (Csato and Opper 2002; Lawrence et al. 2002; Seeger et al. 2003; Quiñonero-Candela and Rasmussen 2005; Snelson and Ghahramani 2006; Titsias 2009; Hensman et al. 2013; Bui et al. 2017) together with stochastic variational inference (Hoffman et al. 2013). Specifically, by using the sparse GP variational inference framework which employs inducing variables (Titsias 2009) as well as its stochastic and non-Gaussian likelihood variants (Hensman et al. 2013; Lloyd et al. 2015; Hensman et al. 2015; Dezfouli and Bonilla 2015; Sheth et al. 2015), we derive an algorithm that can scale to arbitrarily large numbers of data instances. More precisely, the main techniques we introduce regarding scalable GPs are: (i) stochastic variational inference that allows us to train the GP multi-label model by sub-sampling both data instances and labels, and (ii) optimization of the inducing inputs, using gradient-based methods, in extremely high dimensional input spaces involving possibly hundreds of thousands of dimensions; we show that such optimization can significantly improve predictive performance.

It is also noted that the main goal of the paper is to present a Bayesian non-parametric model that is able to scale well to extreme dimensions while at the same time achieves performance similar to other state-of-the-art methods that make use of non-probabilistic models. The remainder of the paper has as follows: Sect. 2 provides a brief discussion about related work. Section 3 describes the form of the multi-label GP model while Sect. 4 discusses scalable variational inference for sparse GP models. Section 5 demonstrates the method using a series of large scale multi-label datasets. Finally, the paper gives some performance characteristics of our method in Sect. 6 concludes with a discussion in Sect. 7.

2 Related work

In the last decade, a multitude of methods have been developed to tackle multi-label classification problems (Tsoumakas and Katakis 2007; Zhang and Zhou 2013). Algorithms such as multi-label random forest (Kocev et al. 2007) or multi-label k-nearest neighbours (Zhang and Zhou 2007) have achieved superior performances than other methods. Nevertheless, all the aforementioned methods fail to scale with the large dimensions describing the eXtreme Multi-label Learning (XML). The last few years, all the proposed XML algorithms fall mainly into two main categories: Label-embedding methods, Tree-based methods.

Label-embedding methods are based on the assumption that the label space can be efficiently approximated by a low-rank structure. The first methods that tried to apply this concept were WSABIE (Weston et al. 2011) and LEML (Yu et al. 2014). Nonetheless, the low-rank representation of the output space failed to lead to high performance due to the information loss followed by the long tail distribution of the labels. This hindrance was later circumvented by SLEEC (Bhatia et al. 2015) and AnnexML (Tagami 2017) where a number of local low-rank embeddings are trained separately based on a partition of the feature space.

Tree-based methods (Prabhu and Varma 2014; Jain et al. 2016; Jasinska et al. 2016; Niculescu-Mizil and Abbasnejad 2017; Si et al. 2017; Siblini et al. 2018; Prabhu et al. 2018; Wydmuch et al. 2018; Khandagale et al. 2020) can be seen as transformation methods that aim to divide the initial large-scale problem into a multiple small-scale sub-problems by recursively partitioning the feature or label space. Those subsets are connected with the nodes of the trees. These methods are usually known for their fast training and prediction time at the expense of predictive performance.

Despite these two major categories, there are recently developed methods (Yen et al. 2016, 2017; Babbar and Schölkopf 2017) which are based on one-vs-rest strategies and/or linear models. For instance, DiSMEC (Babbar and Schölkopf 2017) achieves state-of-the art performance by minimizing Hamming loss with ℓ_2 regularization while an imposed threshold help to reduce model size. These methods are heavily relied on distributed hardware to reduce training and test time. Deep learning methods have also emerged in the XML field (Nam et al. 2017) which make extensive use of GPU resources and more sophisticated text preprocessing techniques (Liu et al. 2017; You et al. 2019).

A few Bayesian methods have also been proposed (He et al. 2012; Kapoor et al. 2012; Jain et al. 2017; Gaure et al. 2017; Papanikolaou and Tsoumakas 2018), however they cannot be trained on XML datasets and/or they attain poor performance comparing to state-of-the art methods. Our method bears similarities with most of those works. Specifically, it is a latent factor model that probabilistically finds a low rank representation of the label matrix Y conditioned on the inputs X, thus, it is also connected to the label-embedding approaches, however, most of these approaches are not probabilistic, so they cannot easily be incorporated to a larger model (possible handling additional types of observed data) or make use of stochastic or on-line statistical learning algorithms. A probabilistic method that is mostly related to ours is the approach by Jain et al. (2017) who constructed a bilinear latent factor model of Y conditioned on X. Some aspects of our method can be thought of as kernelized GP-based extensions of certain linear latent variables in Jain et al. (2017), with the additional difference that we base inference on variational methods while Jain et al. (2017) use Gibbs sampling or on-line EM.

Finally, a previous GP-based method for multi-label learning was proposed by He et al. (2012) and it was based on the multi-task GP model of Bonilla et al. (2008). This method makes use of basic Nyström-type methods to approximate the full multi-task kernel matrix of size $KN \times KN$ with a matrix of size $KM \times KM$ (N is the number of training data points and $M \ll N$; see next section) which leads to complexity $\mathcal{O}(K^3M^3)$, i.e. cubic with respect to the number of labels K , rendering this method impractical for XML problems. Instead, our method utilizes variational sparse GP methods and stochastic optimization to obtain a fully scalable algorithm having a sub-linear complexity over the class labels and data points.

3 The multi-label GP factor model

Suppose a training dataset $\mathscr{D} = (\mathbf{x}^{(i)}, \mathbf{y}^{(i)})_{i=1}^{N}$ where each $\mathbf{x}^{(i)} \in \mathbb{R}^D$ is the input vector and $\mathbf{y}^{(i)} \in \{-1, 1\}^K$ is the binary vector that indicates the class labels assigned to $\mathbf{x}^{(i)}$, so that $y_k = 1$ indicates presence of the k-th label while $y_k = -1$ indicates absence. We will collectively denote all input vectors by $X \in \mathbb{R}^{N \times D}$ and the binary labels by $Y \in \{-1, 1\}^{N \times K}$ so that rows of these matrices store respective data points. We wish to model these data using a flexible probabilistic model that captures the correlation between different labels. We consider a multi-label extension of the semiparametric latent factor model (SLFM) of Teh et al. (2005) that combines a linear latent variable model with GPs. Specifically, SLFM is a general-purpose multi-output GP model (Teh et al. 2005; Álvarez and Lawrence 2011; Alvarez et al. 2012) that uses a small number of P latent GPs (factors) to generate the K outputs through a linear mapping. The full hierarchical model for generating the training examples is,

$$h_p \sim \mathscr{GP}(0, k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})),\ p = 1, \dots, P, \tag{1}$$

$$\mathbf{f}^{(i)} = \Phi \mathbf{h}^{(i)} + \mathbf{b},\ i = 1, \dots, N, \tag{2}$$

$$\mathbf{y}^{(i)} \sim p(\mathbf{y}^{(i)} | \mathbf{h}^{(i)}) = \prod_{k=1}^{K} \sigma(y_k^{(i)} f_k^{(i)}),\ i = 1, \dots, N, \tag{3}$$

where h_p denotes a latent function drawn from a GP with zero-mean and kernel function $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ that depends on kernel hyperparameters $\boldsymbol{\theta}$ (although $\boldsymbol{\theta}$ is suppressed throughout to keep the notation uncluttered). Further, $\mathbf{h}^{(i)} = [h_1^{(i)} \dots, h_P^{(i)}]^\top \in \mathbb{R}^P$ denotes the vector of all function values evaluated at input $\mathbf{x}^{(i)}$, i.e. $h_p^{(i)} \equiv h_p(\mathbf{x}^{(i)})$, while the parameters $\Phi \in \mathbb{R}^{K \times P}$ and $\mathbf{b} \in \mathbb{R}^K$ correspond to the factor loadings matrix and the bias vector of the linear mapping. By using these parameters the latent vector $\mathbf{h}^{(i)}$ is deterministically mapped into $\mathbf{f}^{(i)} = [f_1^{(i)}, \dots, f_K^{(i)}]^\top \in \mathbb{R}^K$, such that each

$$f_k^{(i)} = \sum_{p=1}^{P} \phi_{kp} h_p^{(i)} + b_k, \tag{4}$$

defines the so-called *utility* score that finally generates the k-th binary label through a sigmoidal/Bernoulli likelihood. Notice that while the labels are conditionally independent given $\mathbf{h}^{(i)}$, they become fully coupled once these variables are integrated out. The full joint distribution is given by

$$\prod_{i=1}^{N} p(\mathbf{y}^{(i)} | \mathbf{h}^{(i)}) \prod_{p=1}^{P} p(\mathbf{h}_p), \tag{5}$$

where $p(\mathbf{h}_p) = \mathscr{N}(\mathbf{h}_p | \mathbf{0}, K_X)$ is an N-dimensional Gaussian distribution induced by evaluating the GP prior at the training inputs X with K_X denoting the kernel or covariance matrix, $[K_X]_{ij} = k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$. An equivalent way to write the above model is by using the concept of kernels for multi-task or vector-valued functions (Bonilla et al. 2008; Álvarez and Lawrence 2011; Alvarez et al. 2012). More precisely, observe that the utility scores $f_k^{(i)}$ that directly interact with the data in (3) follow a GP prior with mean given by the bias b_k (that depends on the label but not on the input) and covariance function

$$\mathrm{Cov}(f_l^{(i)}, f_k^{(j)}) = k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \sum_{p=1}^{P} \phi_{lp}\phi_{kp}. \tag{6}$$

For regression problems with Gaussian likelihoods the above multi-task GP is known as the intrinsic correlation model (Stoyan 1996; Bonilla et al. 2008), a specific case of co-kriging in geostatistics; see Alvarez et al. (2012) for a full review. Here, we use this model for multi-label learning where the tasks correspond to different class labels.

Inference in the above model is very challenging since real applications in multi-label classification involve both very large number of training instances N and very large number of class labels K (Zhang and Zhou 2013; Gibaja and Ventura 2014, 2015). In the next section we propose an efficient variational inference algorithm that combines sparse GPs with stochastic variational inference (Hoffman et al. 2013) and scales as $O(PM^3)$ where $M \ll N$.

4 Scalable variational inference

The approximate inference procedures derived in this section are mainly based on the representation that uses the latent GP vectors $\mathbf{u}_p$ rather than the multi-task kernel representation in (6). The utility scores $f_k^{(i)}$ will only be used to simplify the computations of some final Gaussian integrals. Section 4.1 discusses the variational sparse GP formulation, while Sect. 4.2 shows how doubly stochastic optimization can allow to deal with arbitrarily large N and K.

4.1 Sparse approximation

To deal with large number of training data we consider the variational sparse GP inference framework based on inducing variables introduced by Titsias (2009); see also Matthews et al. (2016) for a measure-theoretic derivation of this method and Bauer et al. (2016) for a useful discussion about its properties. For each latent function h_p we introduce a vector $\mathbf{u}_p \in \mathbb{R}^M$ of function values of h_p evaluated at inputs Z, where for simplicity we take the inputs Z to be shared by all latent GPs. The vector $\mathbf{u}_p$ is often referred to as inducing variables and Z as the inducing or pseudo inputs (Quiñonero-Candela and Rasmussen 2005; Snelson and Ghahramani 2006). In the variational sparse GP method Z plays the role of a variational parameter that can be optimized to improve the approximation. By following Titsias (2009) we augment the joint distribution in (5) with the inducing variables to obtain

$$\prod_{i=1}^{n} p(\mathbf{y}^{(i)}|\mathbf{h}^{(i)}) \prod_{p=1}^{P} p(\mathbf{h}_p|\mathbf{u}_p)p(\mathbf{u}_p). \tag{7}$$

Here, $p(\mathbf{u}_p) = \mathcal{N}(\mathbf{u}_p|\mathbf{0}, K_Z)$ is the marginal GP prior over $\mathbf{u}_p$ and K_Z is the $M \times M$ kernel matrix obtained by evaluating the kernel function at Z while $p(\mathbf{h}_p|\mathbf{u}_p)$ is the conditional GP prior

$$p(\mathbf{h}_p|\mathbf{u}_p) = \mathcal{N}\left(\mathbf{h}_p|K_{XZ}K_Z^{-1}\mathbf{u}_p, K_X - K_{XZ}K_Z^{-1}K_{ZX}\right),$$

where K_{XZ} is the cross-covariance matrix between the training inputs X and the inducing inputs Z while $K_{ZX} = K_{XZ}^\top$. For any value of Z this augmentation does not change the model (e.g. the exact marginal likelihood is invariant to the value of Z), however by applying a certain variational approximation in the space of $(\mathbf{h}_p, \mathbf{u}_p)$ we can both reduce the time complexity and also treat Z as a variational parameter. This is achieved by choosing the approximate distribution to be

$$\mathcal{Q} = \prod_{p=1}^{P} p(\mathbf{h}_p|\mathbf{u}_p)q(\mathbf{u}_p), \tag{8}$$

where $p(\mathbf{h}_p|\mathbf{u}_p)$ is the conditional GP prior that appears also in the joint (7), while $q(\mathbf{u}_p) = \mathcal{N}(\mathbf{u}_p|\boldsymbol{\mu}_p, \Sigma_p)$ is a Gaussian variational distribution over the inducing variables for the p-th latent GP. Hence, $\boldsymbol{\mu}_p \in \mathbb{R}^M$ is a real-valued vector of tunable variational parameters, while $\Sigma_p \in \mathbb{R}^{M\times M}$ is the covariance matrix of the variational distribution which is parametrized by $\frac{M^2+M}{2}$ variational parameters since we only need a $M \times M$ lower triangular matrix L_p to express Σ_p. The corresponding $q(\mathbf{h}_p)$ obtained by the previous choice of $q(\mathbf{u}_p)$ (derived by marginalizing out up from the variational distribution in (8)) is $q(\mathbf{h}_p) = \mathcal{N}(\mathbf{h}_p|\boldsymbol{\mu}_p^h, \Sigma_p^h)$ where

$$\boldsymbol{\mu}_p^h = Q_{XZ}\boldsymbol{\mu}_p, \tag{9}$$

$$\Sigma_p^h = K_X - Q_{XZ}K_{ZX} + R_{XZ}^p R_{XZ}^{p^\top}, \tag{10}$$

with $Q_{XZ} = K_{XZ}K_Z^{-1}$ and $R_{XZ}^p = Q_{XZ}L_p$.

To express the lower bound on the log marginal likelihood $\log p(Y)$ under the variational distribution in (8) we start the derivation as in Titsias (2009) which leads to cancellation of each conditional GP prior $p(\mathbf{h}_p|\mathbf{u}_p)$ and then by following the derivation suitable for scalable and/or non-Gaussian likelihoods (Hensman et al. 2013; Lloyd et al. 2015; Hensman et al. 2015; Dezfouli and Bonilla 2015) we derive the final bound. More specifically, we would like to approximate the true posterior $\mathcal{P} \equiv p(\{\mathbf{h}_p, \mathbf{u}_p\}_{p=1}^P|Y)$ with the variational distribution in (8). The minimization of the KL divergence $\mathrm{KL}[\mathcal{Q}||\mathcal{P}]$ is equivalently expressed as the maximization of the following lower bound on the log marginal likelihood $\log p(Y)$,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{Q}}\left[\log \frac{\prod_{i=1}^N p(\mathbf{y}^{(i)}|\mathbf{h}^{(i)}) \prod_{p=1}^P p(\mathbf{h}_p|\mathbf{u}_p)p(\mathbf{u}_p)}{\prod_{p=1}^P p(\mathbf{h}_p|\mathbf{u}_p)q(\mathbf{u}_p)}\right] \\
&= \mathbb{E}_{\mathcal{Q}}\left[\log \frac{\prod_{i=1}^N p(\mathbf{y}^{(i)}|\mathbf{h}^{(i)}) \prod_{p=1}^P p(\mathbf{u}_p)}{\prod_{p=1}^P q(\mathbf{u}_p)}\right] \\
&= \sum_{i=1}^N \mathbb{E}_{\mathcal{Q}}\left[\log p(\mathbf{y}^{(i)}|\mathbf{h}^{(i)})\right] - \sum_{p=1}^P \mathbb{E}_{\mathcal{Q}}\left[\log \frac{q(\mathbf{u}_p)}{p(\mathbf{u}_p)}\right]
\end{aligned}$$

Since each $\mathcal{Q}$ is given by (8), each term in the second sum simplifies to become an expectation over $\mathbf{u}_p$,

$$\mathbb{E}_{\mathcal{Q}}\left[\log \frac{q(\mathbf{u}_p)}{p(\mathbf{u}_p)}\right] = \mathbb{E}_{q(\mathbf{u}_p)}\left[\log \frac{q(\mathbf{u}_p)}{p(\mathbf{u}_p)}\right]$$

which is precisely the KL divergence $\mathrm{KL}[q(\mathbf{u}_p)||p(\mathbf{u}_p)]$. Therefore, the derived bound can be written as

$$\mathcal{F} = -\sum_{i=1}^{N}\sum_{k=1}^{K}\mathbb{E}_{q(f_k^{(i)})}\left[\log(1 + e^{-y_k^{(i)}f_k^{(i)}})\right] - \sum_{p=1}^{P}\mathrm{KL}[q(\mathbf{u}_p)||p(\mathbf{u}_p)]. \tag{11}$$

In the first line of this expression we have written the expectation of each log-likelihood term as an integral under the scalar utility $f_k^{(i)} = \sum_{p=1}^{P}\phi_{kp}h_p^{(i)} + b_k$, defined in (4), that follows the univariate variational Gaussian distribution

$$q(f_k^{(i)}) = \mathcal{N}\left(f_k^{(i)} | \sum_{p=1}^{P}\phi_{kp}\mu_p^{(i)} + b_k, \sum_{p=1}^{P}\phi_{kp}^2 s_p^{(i)}\right), \tag{12}$$

where $\mu_p^{(i)}$ is the i-th element of the vector $\boldsymbol{\mu}_p^h$ defined in (9) and $s_p^{(i)}$ the i-th diagonal element (i.e. variance) of the covariance matrix Σ_p^h from (10). Clearly, all expectations over the likelihood terms reduce to performing NK one-dimensional integrals under Gaussian distributions and each such integral can be accurately approximated by Gauss-Hermite quadrature.

Each KL divergence term of the lower bound in the second line of Eq. (11) is given by

$$\mathrm{KL}[q(\mathbf{u}_p)||p(\mathbf{u}_p)] = \frac{1}{2}\boldsymbol{\mu}_p^\top K_Z^{-1}\boldsymbol{\mu}_p + \frac{1}{2}\operatorname{tr}\left(K_Z^{-1}\Sigma_p\right) + \frac{1}{2}\log|K_Z| - \frac{1}{2}\log|\Sigma_p| - \frac{M}{2}. \tag{13}$$

Notice that this term and the overall bound in (11) can be computed efficiently while numerical stability can be achieved by a ”jitter” addition on K_Z. More specifically, we only need to compute once the Cholesky decomposition of K_Z in order to evaluate the P KL divergences, while the fact that we only keep the lower triangular matrix L_p of each variational covariance matrix, allows us to efficiently calculate all the remaining terms of each KL divergence.

To compute the bound we need firstly to perform one Cholesky decomposition of the matrix K_Z that overall scales as $\mathcal{O}(M^3)$ and allows us to fully calculate the sum of the KL divergence terms in the second line in (11). Then, with this Cholesky decomposition precomputed, for each i-th data point we need to compute $(\mu_p^{(i)}, s_p^{(i)})_{p=1}^P$, an operation that scales as $\mathcal{O}(PM^2)$, and subsequently calculate the K variational distributions (i.e. their means and variances) over the utility scores in (12) which requires additional $\mathcal{O}(KP)$ time. Therefore, in order to compute the whole data reconstruction term of the bound (first line in Eq. (11)) we need $\mathcal{O}(NKP + NPM^2)$ time and for the full bound we need $\mathcal{O}(NKP + NPM^2 + M^3)$ time. Given that $N \gg M$ and $K \gg P$, the terms that can dominate are either $\mathcal{O}(NKP)$ or $\mathcal{O}(NPM^2)$ which can make the computations very expensive when the number of data instances and/or labels is very large. Next, we show how to make the optimization of the bound scalable for arbitrarily large numbers of data points and labels.

4.2 Scalable training using stochastic optimization

To ensure that the time complexity $\mathcal{O}(NKP + NPM^2 + M^3)$ for very large datasets is reduced to $\mathcal{O}(PM^3)$ we shall optimize the bound using stochastic gradient ascent by following a similar procedure used in stochastic variational inference for GPs Hensman et al.

(2013). Given that the sum of KL divergences in (11) is already within the desired complexity $\mathcal{O}(PM^3)$, we only need to speed up the remaining data reconstruction term. This term involves a double sum over data instances and class labels, a setting suitable for stochastic approximation. Thus, a straightforward procedure is to uniformly sub-sample terms in the double sum in (11) which leads to an unbiased estimate of the bound and its gradients. It turns out that we can further reduce the variance of this basic strategy by applying a more stratified sub-sampling over class labels as discussed next.

Suppose $\mathcal{B} \subset \{1, \dots, N\}$ denotes the current minibatch at the t-th iteration of stochastic gradient ascent. For each $i \in \mathcal{B}$ the internal sum over class labels can be written as

$$-\sum_{k \in \mathcal{P}_i} \mathbb{E}_{q(f_k^{(i)})}\left[\log(1 + e^{-f_k^{(i)}})\right] - \sum_{\ell \in \mathcal{N}_i} \mathbb{E}_{q(f_\ell^{(i)})}\left[\log(1 + e^{f_\ell^{(i)}})\right],$$

where $\mathcal{P}_i = \{k | y_k^{(i)} = 1\}$ is the set of present or positive labels of $\mathbf{x}^{(i)}$ while $\mathcal{N}_i = \{k | y_k^{(i)} = -1\}$ is the set of absent or negative labels such that $\mathcal{P}_i \cup \mathcal{N}_i = \{1, \cdots, K\}$. In typical multi-label classification problems (Zhang and Zhou 2013; Gibaja and Ventura 2014, 2015) the size of positive labels $\mathcal{P}_i$ is very small, while the negative set can be extremely large. Thus, we can enumerate exactly the first sum and use (if needed) sub-sampling to approximate the second sum over the negative labels. The whole process becomes somehow similar to negative sampling used in large scale classification and for learning word embeddings Mikolov et al. (2013). Overall, we get the following unbiased stochastic estimate of the lower bound,

$$-\frac{N}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left(\sum_{k \in \mathcal{P}_i} \mathbb{E}_{q(f_k^{(i)})}\left[\log(1 + e^{-f_k^{(i)}})\right] + \frac{|\mathcal{N}_i|}{|\mathcal{L}_i|} \sum_{\ell \in \mathcal{L}_i} \mathbb{E}_{q(f_\ell^{(i)})}\left[\log(1 + e^{f_\ell^{(i)}})\right] \right)$$
$$- \sum_{p=1}^{P} \mathrm{KL}[q(\mathbf{u}_p) || p(\mathbf{u}_p)],$$

where $\mathcal{L}_i$ is the set of negative classes for the i-th data point. In general, the computation of this stochastic bound scales as $\mathcal{O}(|\mathcal{B}|(|\mathcal{P}_i| + |\mathcal{L}_i|)P + |\mathcal{B}|PM^2 + M^3)$ and by choosing $|\mathcal{B}| \sim \mathcal{O}(M)$ and $|\mathcal{P}_i| + |\mathcal{L}_i| \sim \mathcal{O}(M^2)$ we can ensure that the overall time is $\mathcal{O}(PM^3)$. Notice that the second condition is not that restrictive and in many cases might not be needed, i.e. in practice we can use very large negative sets $\mathcal{L}_i$ which for many datasets could be equal to the full negative set $\mathcal{N}_i$.

We implemented the above stochastic bound in Python (where the one-dimensional integrals are obtained by Gauss-Hermite quadrature) in order to jointly optimize using stochastic gradient ascent and automatic differentiation tools[1] over the parameters $(\mathbf{\Phi}, \mathbf{b})$ of the linear mapping, the $\frac{PM(M+3)}{2}$ variational parameters $\{\boldsymbol{\mu}_p, L_p\}_{p=1}^{P}$ of the variational distributions $q(\mathbf{u}_p)$, the inducing inputs Z and the kernel hyperparameters $\boldsymbol{\theta}$.

4.3 Prediction

Given a novel data point $\mathbf{x}^{(*)}$ we would like to make prediction over its unknown label vector $\mathbf{y}^{(*)}$. This requires approximating the predictive distribution $p(\mathbf{y}^{(*)}|Y)$,

[1] We use tensorflow: https://www.tensorflow.org/.

Table 1 Data sets statistics: N and N_{ts} are the number of the training and test points respectively, D and K are the number of features and labels respectively, and $\overline{K}$ is the average number of positive labels in an instance

Data set	D	K	N	N_{ts}
Bibtex	1836	159	4880	2515
Delicious	500	983	12,920	3185
EUR-Lex	5000	3993	15,539	3809
Wiki10	101,938	30,938	14,146	6616
AmazonCat	203,882	13,330	1,186,239	306,782
Delicious-large	782,585	205,443	196,606	100,095

$$p(\mathbf{y}^{(*)}|Y) \approx \int p(\mathbf{y}^{(*)}|\mathbf{u}^{(*)})q(\mathbf{u}^{(*)})d\mathbf{u}^{(*)}. \tag{14}$$

Here, $q(\mathbf{u}^{(*)})$ is the variational predictive posterior over the latent function values $\mathbf{u}^{(*)}$ evaluated at $\mathbf{x}^{(*)}$. An interesting aspect of the variational sparse GP method is that to obtain $q(\mathbf{u}^{(*)})$ we need to make no further approximations since everything follows from the GP consistency property, i.e.

$$\begin{aligned} q(\mathbf{u}^{(*)}) &= \prod_{p=1}^{P} \int p(u_p^{(*)}|\mathbf{h}_p, \mathbf{u}_p)p(\mathbf{h}_p|\mathbf{u}_p)q(\mathbf{u}_p)d\mathbf{h}_p d\mathbf{u}_p \\ &= \prod_{p=1}^{P} \int p(u_p^{(*)}|\mathbf{u}_p)q(\mathbf{u}_p)d\mathbf{u}_p = \prod_{p=1}^{P} q(u_p^{(*)}). \end{aligned} \tag{15}$$

Here, GP consistency tractably simplifies each integral $\int p(u_p^{(*)}|\mathbf{h}_p, \mathbf{u}_p)p(\mathbf{h}_p|\mathbf{u}_p)d\mathbf{h}_p$ to $p(u_p^{(*)}|\mathbf{u}_p)$ so that the obtained $p(u_p^{(*)}|\mathbf{u}_p)$ is the conditional GP prior of $u_p^{(*)}$ given the inducing variables. The final form of each univariate Gaussian $q(u_p^{(*)})$ has a mean and variance given precisely by Eqs. (9) and (10) with X replaced by $\mathbf{x}^{(*)}$. In practice, when we compute several accuracy ranking-based scores that are often used in the literature to report multi-label classification performance Zhang and Zhou (2013), Gibaja and Ventura (2014), Gibaja and Ventura (2015) it suffices to further approximate $q(\mathbf{u}^{(*)})$ by a delta mass centred at the MAP[2]. This reduces the whole computation of such scores to only requiring the evaluation of the mean utility vector $\bar{\mathbf{f}}^{(*)} = [\bar{f}_1^{(*)} \dots \bar{f}_K^{(*)}]^\top$ such that $\bar{f}_k^{(*)} = \sum_{p=1}^{P} \phi_{kp}\mu_p^{(*)} + b_k$, where $\mu_p^{(*)} = \mathbf{k}(\mathbf{x}^{(*)}, Z)K_Z^{-1}\boldsymbol{\mu}_p$ and $\mathbf{k}(\mathbf{x}^{(*)}, Z)$ is the cross covariance row vector between $\mathbf{x}^{(*)}$ and the inducing points Z. By using $\bar{\mathbf{f}}^{(*)}$ we can compute several ranking scores as described in the experiments section after.

5 Experiments

Experiments are carried out on three small-scale (*SS*) real-world datasets, Bibtex Katakis et al. (2008), Delicious Tsoumakas et al. (2008), and EUR-Lex Mencia and Fürnkranz (2008) and three large-scale (*LS*) real-world datasets, Wiki10 Zubiaga (2012), AmazonCat McAuley et al. (2015), and Delicious-L(arge) Wetzker et al. (2008). All the datasets are

[2] Estimating such accuracy scores using a more accurate Monte Carlo estimation of (14) leads to very similar results.

publicly available; see Table 1 for summary statistics of each dataset. To the best of our knowledge, this is the first time that a GB-based method is applied to datasets of that large dimensions in the context of multi-label learning. Nevertheless, datasets of even larger dimensions such as WikiLSHTC325K (Partalas et al. 2015) or Amazon3M (McAuley et al. 2015) were not able to be trained by our framework due to hardware limitations. The current code is available at https://github.com/aresPanos/mlgpf.

In our experiments, we applied the proposed multi-label GP factor model (MLGPF) using a linear and a squared exponential (SE) kernel, where in both cases we freely optimized the inducing inputs matrix Z resulting to methods *LR* and *SE* respectively. It is also worth mentioning that for those two kernels, we add D extra tunable hyperparameters – one for each input dimension – in order to improve our model performance in the extremely high dimensional datasets. We also employed a linear combination of those two kernels for providing extra flexibility to our model. Initialization of Z is achieved by running a few iterations of the k-means algorithm while the rest of the variational parameters are initialised by i.i.d. standard Gaussian samples. Additionally, we consider the case where the inducing inputs are randomly chosen from the training instances and are kept fixed throughout the optimization process. This is to demonstrate the crucial role of optimizing inducing inputs regarding the posterior approximation and the predictive performance. For this case, we employ the SE kernel and we denote that method by *SE-F*. Further, it should be mentioned that choosing large size for the negative label set $\mathscr{L}_i$ can dramatically reduce variance in the stochastic optimization of the lower bound. Fortunately, the computational complexity analysis of Sect. 4.2 allows us to choose $\mathscr{L}_i$ such that $|\mathscr{L}_i| \sim \mathscr{O}(M^2)$ which is not that restrictive, and in practice, for all datasets used in the experiments, $\mathscr{L}_i$ is selected to be equal to the size of the full negative set $\mathscr{N}_i$. For the evaluation of the one-dimensional integrals by Gauss-Hermite method in (11) we use ten quadrature points which allows to have a sufficiently close approximation without any increase on the computational cost of the algorithm.

For the small-scale datasets we run the MLGPF model ten times using ten different random initializations and we compute the corresponding average precision scores (see Table 7) and lower bounds (see Fig. 5) accompanied by their corresponding standard deviations. The high computational cost and training time of the large-scale datasets restricts us to just a single run for the remaining three large datasets.

We evaluate the predictive performance of our method against some of the state-of-the-art algorithms by using the *Precision@k* score (P@k). For a ground truth test vector $\mathbf{y}^{(*)} \in \{-1, 1\}^K$ and a predicted score vector $\bar{\mathbf{f}}^{(*)} \in \mathbb{R}^K$, the P@k is defined as $k^{-1} \sum_{l \in \mathrm{rank}_k(\bar{\mathbf{f}}^{(*)})} (\mathbf{y}_l^{(*)} + 1)/2$, where $\mathrm{rank}_k(\bar{\mathbf{f}}^{(*)})$ returns indices of the k largest values of $\bar{\mathbf{f}}^{(*)}$ in descending order. Here, $\bar{\mathbf{f}}^{(*)}$ can be evaluated using the trained MLGPF model as described in Sect. 4.3. Such ranking-based evaluation of multi-label models is very standard in the multi-label literature where we only care to predict the few 1s in the multi-label vectors, typical in many real-world examples, such as recommender systems; for example, see Prabhu and Varma (2014); Jain et al. (2017) and the reported results in the *Extreme Classification Repository*.[3] Moreover, following common practice in the extreme label literature we examine and compare the robustness of our method against tail labels, that is labels that appear rarely in the dataset. This is achieved by using the propensity score version of P@k, namely PSP@k, proposed in Jain et al. (2016). Finally, for completeness, we

[3] http://manikvarma.org/downloads/XC/XMLRepository.html

Table 2 Parameter settings of the MLGPF model for each dataset used in our experiments

Dataset	Bibtex	Delicious	EUR-Lex	Wiki10	AmazonCat	Delicious-L
P	159	300	500	700	600	600
M	400	400	400	600	600	400

Table 3 Training time (in hours) comparison between the MLGPF model and Parabel/ProXML over the six multi-label datasets

SS	Bibtex	Delicious	EUR-Lex
MLGPF	1.1	2.88	15.24
ProXML	0.03	0.25	4.17
Parabel	0.003	0.02	0.06
LS	Wiki10	AmazonCat	Delicious-L
MLGPF	120.79	495.28	478.61
ProXML	99.39	447.65	-
Parabel	0.68	1.01	17.61

For MLGPF model, we make use of a linear combination of *SE* and *LR* kernel. Absence of results due to high computational time is denoted by '–'

also report results for nDCG performance measure and its propensity version PSnDCG@k, see Jain et al. (2016) (Fig. 1).

The parameter settings for each dataset can be found in Table 2. Parameters are primarily chosen in that way such that the memory footprint is not large enough for the 64Gb RAM Intel Xeon E5-2686 v4 @ 2.30GHz on which we run the experiments. Nevertheless, the aforementioned limitations do not impose much restriction on the performance of our model. A detailed comparison of the effect that different values of *P* and *M* have on our model using the Bibtex dataset can be found in Figs. 3 and 4 while Fig. 2 show the importance of increasing the values of *P* and *M* for the general performance of the model.

Based on both predictive performance (see Table 7) and computational time (see Table 3) of all three methods on small-scale datasets, we choose to solely employ the *SE* for large-scale datasets (except Wiki10, where *SE-F* is also used), since it combines high predictive power and low computational cost; attributes very important for the challenging task of training our model on those datasets.

Furthermore, we compare the MLGPF model with a linear combination of *SE* and *LR* kernels and optimized inducing inputs with six state-of-the-art-methods from the literature, namely SLEEC Bhatia et al. (2015), PFastreXML Jain et al. (2016), PD-Sparse Yen et al. (2016), DiSMEC Babbar and Schölkopf (2017), Parabel Prabhu et al. (2018), and ProXML Babbar and Schölkopf (2019) as they are reported in the *Extreme Classification Repository (ECR)* (see footnote 3). Results that do not appear in ECR were generated by using the publicly available codes with the suggested configurations. All those results appear in Tables 4 and 5 for P@k and PSP@k, respectively. Additional results, including nDCG@k and PSnDCG@k are available in Table 6. The choice of the specific methods is based on the high predictive power on both small and large scale datasets. At this point we would like to note that despite the similarities of our work with

Table 4 Performance comparison between the MLGPF model, where a linear combination of *SE* and *LR* is used, and other state-of-the-art methods

Dataset		MLGPF	SLEEC	PfastreXML	PD-Sparse	Parabel	DiSMEC	ProXML
Bibtex	P@1	**66.51**	**65.08**	63.46	61.29	64.53	63.69	64.60
	P@3	**41.12**	**39.64**	39.22	35.82	38.56	38.80	39.00
	P@5	**30.34**	28.87	**29.14**	25.74	27.94	28.30	28.20
Delicious	P@1	**69.09**	**67.59**	67.13	51.82	67.44	65.79	65.92
	P@3	**63.27**	61.38	**62.33**	44.18	61.83	62.06	61.32
	P@5	**58.61**	56.56	**58.62**	38.95	56.75	58.53	56.39
EUR-Lex	P@1	**82.45**	79.26	75.45	76.43	81.73	82.40	**83.41**
	P@3	68.39	64.30	62.70	60.37	**68.78**	68.50	**70.96**
	P@5	56.31	52.33	52.51	49.72	57.44	**57.70**	**58.92**
Wiki10	P@1	84.92	**85.88**	83.57	82.69	84.31	**85.20**	82.48
	P@3	71.75	**72.98**	68.61	67.13	72.57	**74.60**	70.73
	P@5	62.22	**62.70**	59.10	58.13	63.39	**65.90**	61.47
AmazonCat	P@1	**92.55**	90.53	91.75	90.60	93.03	93.40	92.15
	P@3	77.32	76.33	77.97	75.14	**79.16**	79.10	78.02
	P@5	62.34	61.52	63.68	60.69	**64.52**	64.10	63.95
Delicious-L	P@1	43.27	**47.85**	41.72	34.37	**46.97**	45.50	–
	P@3	**40.21**	**42.21**	37.83	29.48	40.08	38.70	–
	P@5	**38.02**	**39.43**	35.58	27.04	36.63	35.50	–

The top-2 P@k (k=1,3,5) for each dataset are in bold. Absence of results due to high computational time is denoted by '–'

Moreno-Muñoz et al. (2018) as we mention at the beginning, the code provided by the latter is not capable to deal with the extremely large number of input and label dimensions and cannot be deployed for that kind of problems, rendering our tensorflow-based implementation more suitable. We also acknowledge that deep-learning based methods, see for example You et al. (2019), are known for their state-of-the-art accuracy results but they are heavily relied on sophisticated text pre-processing techniques and advanced computing hardware, and thus, we omit any comparisons with them.

Regarding predictive performance of the MLGPF model, it achieves the best P@k scores in Bibtex and Delicious dataset compared to the other state-of-the-art methods. Additionally, the predictive ability in each of the remaining datasets, even for the large-scale ones, is very close to the best state-of-the-art method, rendering our model highly competitive overall. For instance, in Eurlex the P@1 score of our model exceeds more than 7% the PFastreXML method while in the largest dataset in our experiments, Delicious-L, MLGPF attains a P@5 score very close to the SLEEC, surpassing in that way all the remaining methods. Similar patterns can be seen in Table 5, where our methods exhibits robustness against tail labels with respect to PSP@k measure.

A detailed training time comparison between our method and ProXML/Parabel is given in Table 3. The results indicate that the Bayesian nature of our framework and the large number of parameters to be optimized pay its toll when it comes to computational speed.

Finally, Fig. 1 and Table 7 reveal the positive correlation that the lower bound has with the predictive performance, while at the same time it is stressed the significance

Table 5 Performance comparison, using the PSP@k metric, between the MLGPF model, where a linear combination of *SE* and *LR* is employed, and other state-of-the-art methods

Dataset		MLGPF	SLEEC	PfastreXML	PD-Sparse	Parabel	DiSMEC	ProXML
Bibtex	PSP@1	**52.95**	51.12	**52.28**	48.34	50.88	49.55	50.1
	PSP@3	**55.27**	53.95	**54.36**	48.77	52.42	52.87	52.0
	PSP@5	**61.36**	59.56	**60.55**	52.93	57.36	59.14	58.3
Delicious	PSP@1	**34.92**	32.11	**34.57**	25.22	32.69	31.37	32.24
	PSP@3	**35.89**	33.21	**34.80**	24.63	34.00	32.90	33.64
	PSP@5	**35.95**	33.83	**35.86**	23.85	34.53	33.33	34.01
EUR-Lex	PSP@1	41.25	34.25	**43.86**	38.28	36.36	41.20	**44.83**
	PSP@3	**45.81**	39.83	45.72	42.00	44.04	45.40	**48.31**
	PSP@5	47.58	42.76	46.97	44.89	48.29	**49.30**	**50.69**
Wiki10	PSP@1	**15.33**	11.14	**19.02**	10.58	11.66	13.60	11.31
	PSP@3	**15.39**	11.86	**18.34**	11.02	12.73	13.10	11.14
	PSP@5	**15.43**	12.40	**18.43**	12.21	13.68	13.80	11.40
AmazonCat	PSP@1	56.28	46.75	**69.52**	49.58	50.93	59.10	**61.92**
	PSP@3	63.57	58.46	**73.22**	61.63	64.00	**67.10**	65.88
	PSP@5	71.75	65.96	**75.48**	68.23	72.08	71.20	**73.58**
Delicious-L	PSP@1	6.85	**7.17**	3.15	5.29	**7.25**	6.5	–
	PSP@3	7.89	**8.16**	3.87	5.80	**7.94**	7.6	–
	PSP@5	**8.72**	**8.96**	4.43	6.24	8.52	8.4	–

The top-2 PSP@k (k=1,3,5) for each dataset are in bold. Absence of results due to high computational time is denoted by '–'

Table 6 Predictive performance of the MLGPF model for the six multi-label datasets with respect to the nDCG@k and PSnDCG@1 scores

Dataset	Bibtex	Delicious	EUR-Lex	Wiki10	AmazonCat	Delicious-L
nDCG@1	66.51	69.09	82.45	84.92	92.55	43.27
nDCG@3	61.67	64.40	72.13	74.74	85.72	41.47
nDCG@5	64.24	60.87	66.07	67.26	84.45	38.95
PSnDCG@1	52.95	34.92	41.25	15.33	56.28	6.85
PSnDCG@3	55.07	35.49	44.66	15.38	62.18	7.42
PSnDCG@5	58.67	35.64	45.98	15.40	70.98	8.28

of optimizing over the inducing inputs Z (especially when the input dimensionality is increased). For instance, the lower bound of the *SE* for EUR-Lex dataset is significantly higher than the bound of *SE-F*, as shown in Fig. 1a, which leads to considerably better P@k scores for *SE* against *SE-F*.

Table 7 Predictive Performance of the MLGPF model for the six multi-label datasets

SS	Bibtex	Delicious	EUR-Lex
LR	66.07 ± 0.09	66.72 ± 0.23	81.13 ± 0.19
	40.68 ± 0.11	61.09 ± 0.16	66.83 ± 0.15
	29.90 ± 0.04	55.94 ± 0.09	55.13 ± 0.05
SE	**66.46 ± 0.12**	**69.02 ± 0.16**	**82.48 ± 0.13**
	41.05 ± 0.08	**63.22 ± 0.11**	**68.43 ± 0.06**
	30.26 ± 0.06	**58.64 ± 0.07**	**56.54 ± 0.04**
SE-F	63.04 ± 0.13	66.43 ± 0.29	75.32 ± 0.17
	39.27 ± 0.11	61.24 ± 0.20	62.07 ± 0.12
	29.11 ± 0.06	56.83 ± 0.11	51.21 ± 0.08
LS	Wiki10	AmazonCat	Delicious-L
SE	83.90	92.48	42.61
	70.73	77.21	39.42
	61.00	62.16	37.42

For the small scale datasets we provide the mean precision values along with standard deviations over ten random initializations where the bold scores are the top ones across the three methods (left-most column). The three rows for each method represent the values (in tandem with the corresponding standard deviations) of P@1, P@3, and P@5

6 Performance characteristics

Table 8 presents a detailed comparison of the effect that different values of the number of GPs P and number of inducing inputs M have on the optimization of the lower bound and the predictive performance of the MLGPF model. Lower bounds for the small scale datasets along with 95% confidence intervals are also provided; see Table 8.

6.1 Extra experimental results

Figures 3, 4, and 5 depict various comparisons in terms of predictive performance and optimization of the lower bound for the Bibtex dataset. Figure 3 shows the evolution of the P@1 throughout epochs for different values of P and M while Fig. 4 presents the same information as previously but in terms of the values of the lower bound. In all the above figures, the *SE* method is used for all cases. The last Figure gives the lower bounds for the small scale datasets in tandem with their corresponding 95% confidence intervals based on the experiments of the main paper.

7 Discussion

We have presented a GP factor model for multi-label learning and we have applied it to real-world large scale datasets involving hundreds of thousands input and label dimensions with similar or even higher predictive performance than other state-of-the-art methods. To achieve scalability, we have introduced a doubly stochastic variational

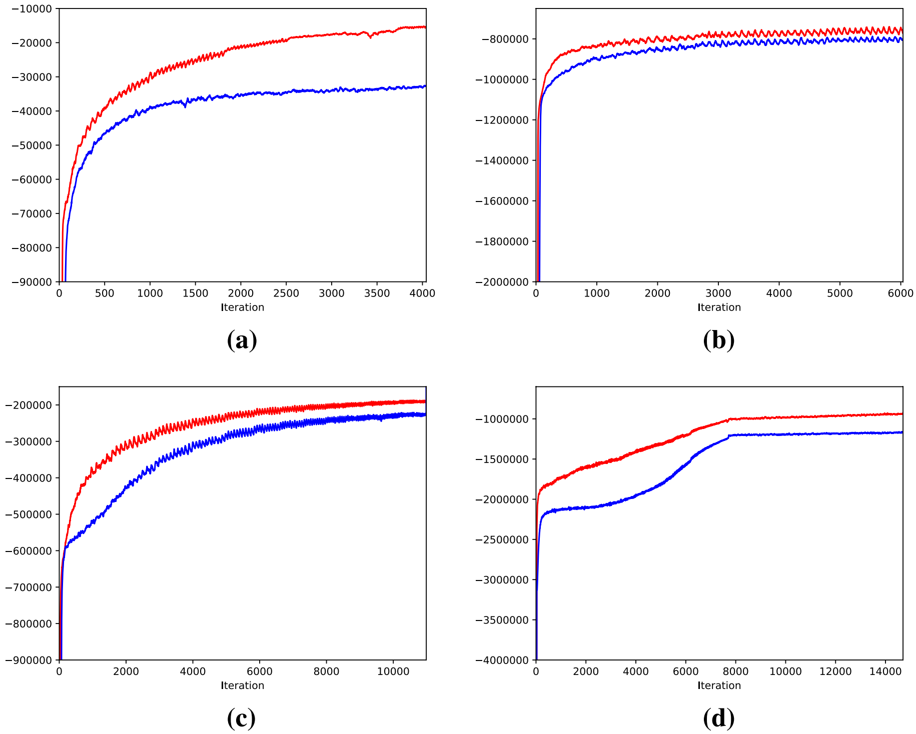


Fig. 1 Evolution of the lower bound for **a** Bibtex, **b** Delicious, **c** EUR-Lex, and **d** Wiki10. The blue line corresponds to maximization of the lower bound using fixed inducing inputs *Z*, while the red one to optimized *Z*. The SE kernel is used for each dataset while all the other parameters are set as described in Table 2 (Color figure online)

inference scheme which could be useful to other non GP-based models for multi-label learning, while the simplified variational approximation could be useful to other GP applications, involving non-Gaussian likelihoods, such as standard multi-class GP classification.

Regarding future research, an important computational challenge is concerned with the development of efficient optimization schemes over inducing inputs in extremely high-dimensional input spaces, which is typical in many applications of multi-label learning Prabhu and Varma (2014). For that, we plan to exploit the significant sparsity patterns of these high dimensional points and optimize inducing inputs so that such sparsity patterns are preserved. Furthermore, we wish to elaborate more on the modelling aspects of our method such as, to modify the likelihood in order to deal with missing labels Jain et al. (2017) and add extra latent variables that can capture non-input dependent correlation between the class labels Gibaja and Ventura (2014). Finally, inspired by the encouraging results of the recent work on natural gradient-based optimization for sparse GP models for non-conjugate likelihoods in Salimbeni et al. (2018), we aim to explore its efficiency to our multi-label framework in those high-dimensional regimes.

Fig. 2 Bar plots comparing both the P@1 and the lower bounds for Bibtex dataset using **a**, **c** fixed P={80, 159}, and **b**, **d** fixed M={50, 100}. For all cases the SE kernel is deployed while we run the MLGPF model for 50 epochs

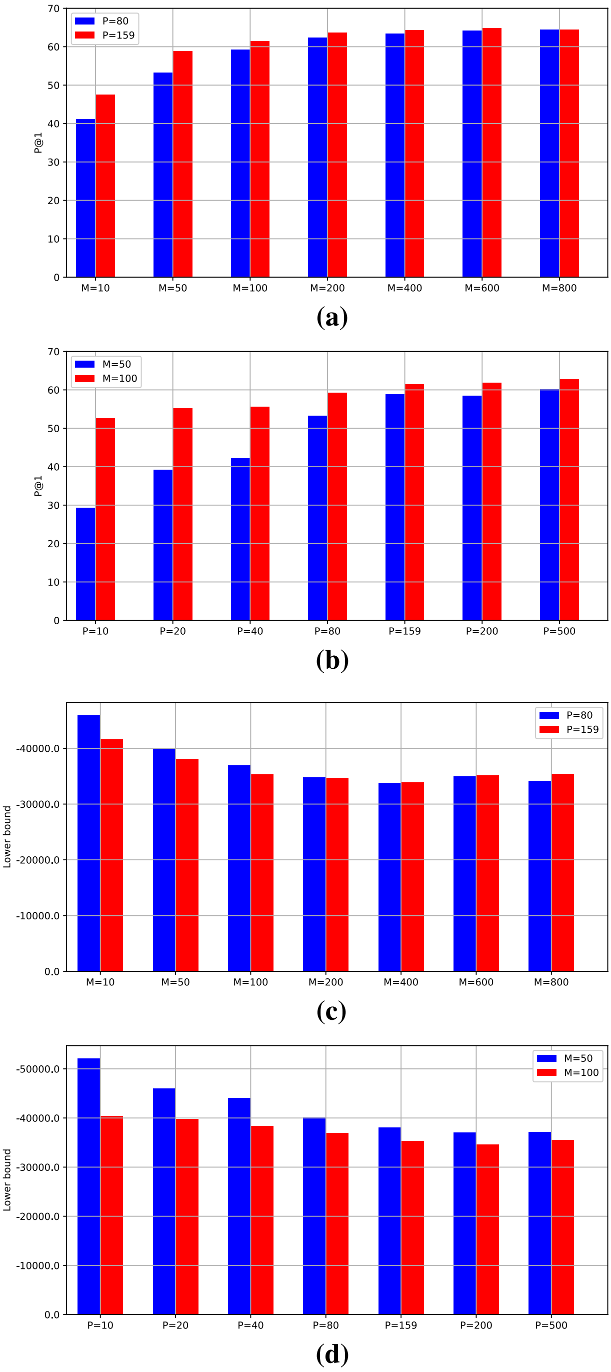
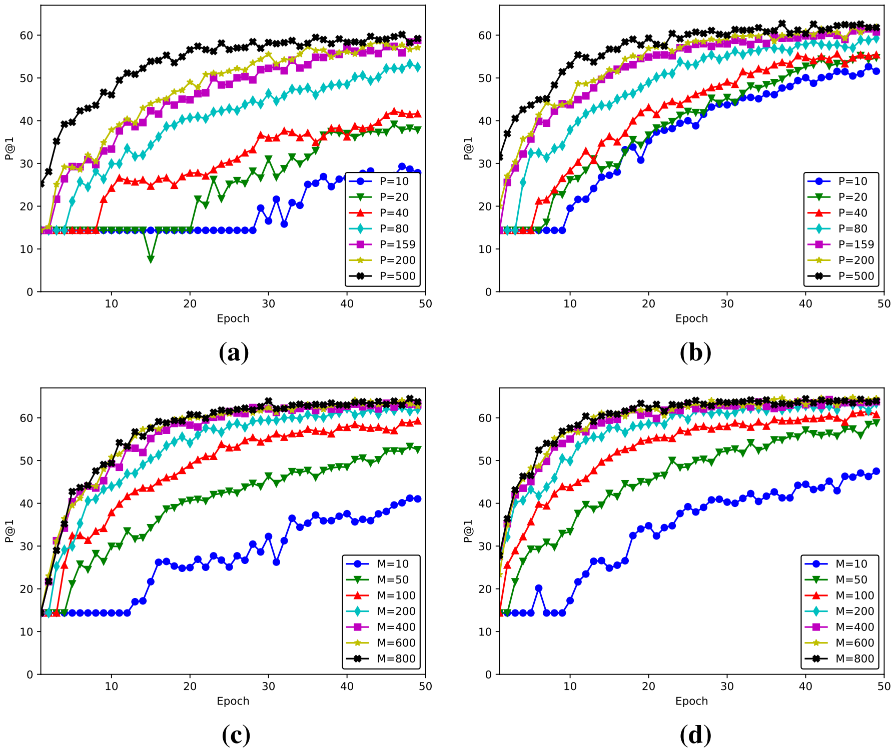


Table 8 Predictive Performance of the MLGPF model for the six multi-label datasets

SS	Bibtex	Delicious	EUR-Lex
LR	66.07 ± 0.09	66.72 ± 0.23	81.13 ± 0.19
	40.68 ± 0.11	61.09 ± 0.16	66.83 ± 0.15
	29.90 ± 0.04	55.94 ± 0.09	55.13 ± 0.05
SE	66.46 ± 0.12	69.02 ± 0.16	82.48 ± 0.13
	41.05 ± 0.08	63.22 ± 0.11	68.43 ± 0.06
	30.26 ± 0.06	58.64 ± 0.07	56.54 ± 0.04
SE-F	63.04 ± 0.13	66.43 ± 0.29	75.32 ± 0.17
	39.27 ± 0.11	61.24 ± 0.20	62.07 ± 0.12
	29.11 ± 0.06	56.83 ± 0.11	51.21 ± 0.08
LS	**Wiki10**	**AmazonCat**	**Delicious-Large**
SE	83.90	92.48	42.61
	70.73	77.21	39.42
	61.00	62.16	37.42
SE-F	77.79	–	–
	64.23	–	–
	55.35	–	–

The three rows for each method (left-most column) represent the values (in tandem with the corresponding standard deviations) of P@1, P@3, and P@5. Those methods that have not been applied on a dataset are indicated with the '–' sign



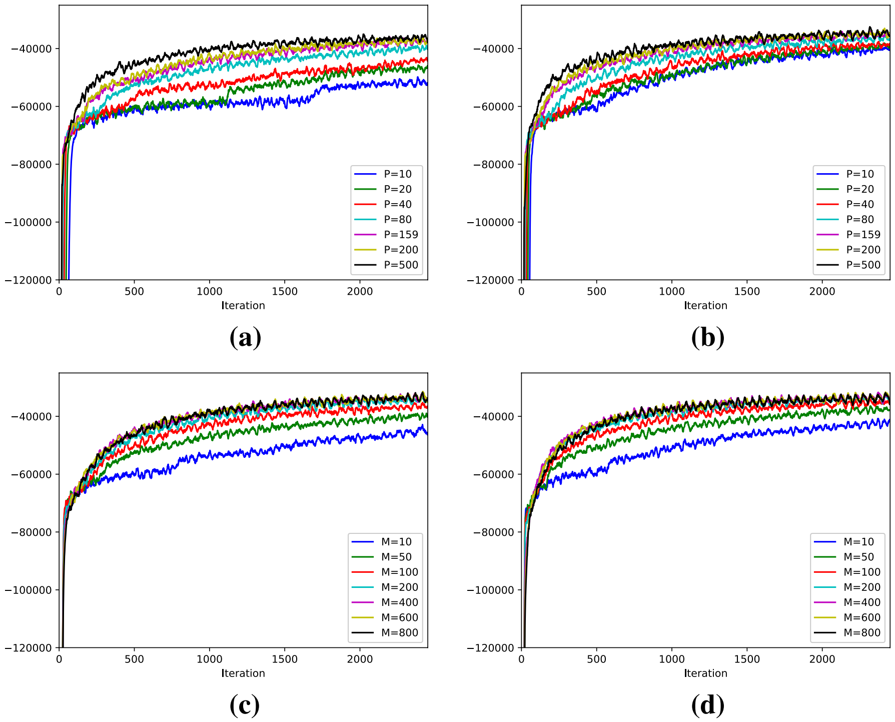
(a)

(b)

(c)

(d)

Fig. 3 P@1 of Bibtex dataset using **a** fixed M = 50, **b** fixed M = 100, **c** fixed P = 80, and **d** fixed P = 159



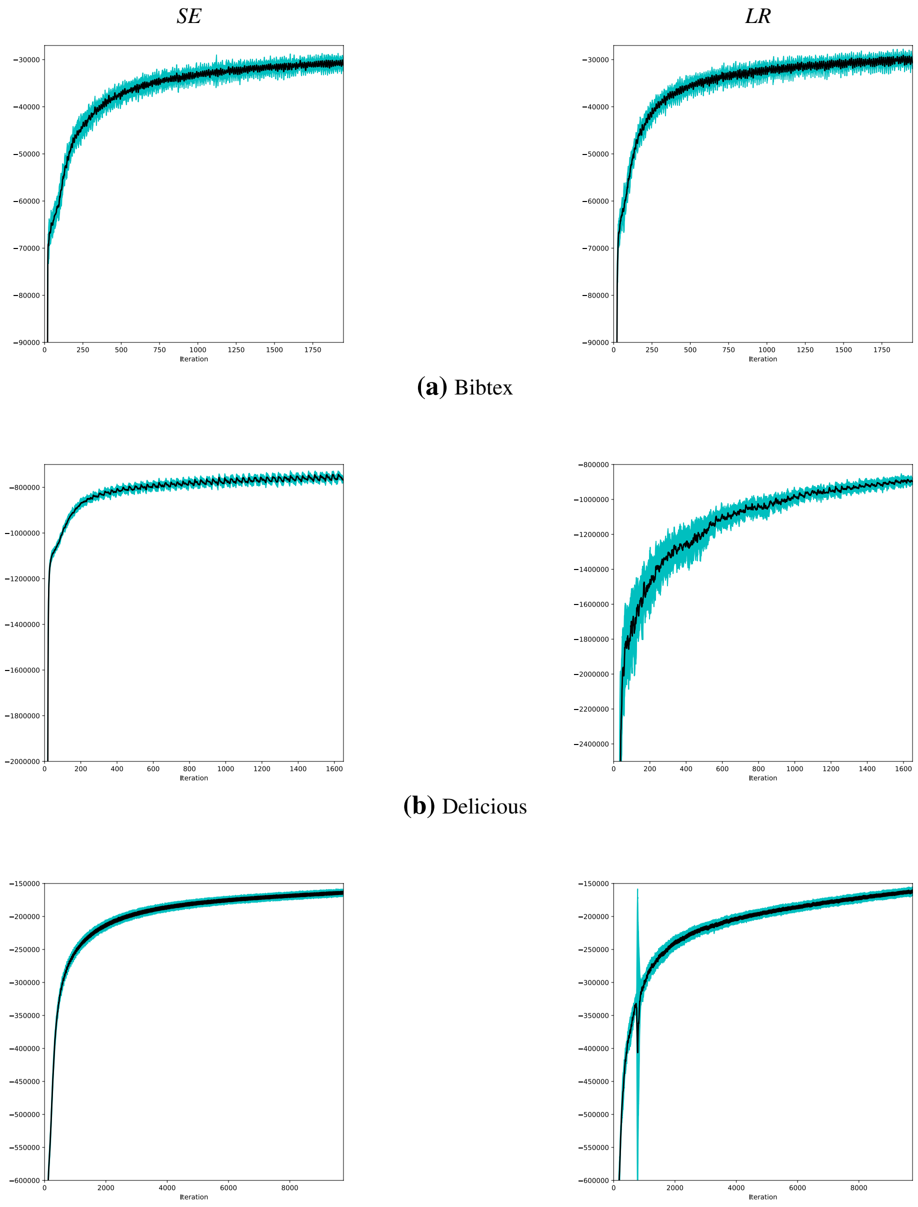
(a)

(b)

(c)

(d)

Fig. 4 Lower bounds of Bibtex dataset using **a** fixed M = 50, **b** fixed M = 100, **c** fixed P = 80, and **d** fixed P = 159



(a) Bibtex

(b) Delicious

(c) EUR-Lex

Fig. 5 Average lower bounds (black line) with 95% confidence intervals (cyan-coloured area) for each of the small-scale datasets based on ten random initializations. Left colums indicates use of the linear kernel while the right one the SE kernel (Color figure online)

Acknowledgements AP and PD were part funded by the The Alan Turing Institute under the Engineering and Physical Sciences Research Council grant EP/N510129/1.

References

Álvarez, M. A., & Lawrence, N. D. (2011). Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning and Research, 12,* 1459–1500.

Álvarez, M. A., Rosasco, L., & Lawrence, N. D. (2012). Kernels for vector-valued functions: a review. *Foundations and Trends R in Machine Learning, 4,*(3) 195–266.

Babbar, R., & Schölkopf, B. (2017) Dismec: distributed sparse machines for extreme multi-label classification. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, pp 721–729.

Babbar, R., & Schölkopf, B. (2019). Data scarcity, robustness and extreme multi-label classification. *Machine Learning, 108*(8–9), 1329–1351.

Bauer, M., van der Wilk, M., & Rasmussen, C.E. (2016). Understanding probabilistic sparse Gaussian process approximations. In: Advances in Neural Information Processing Systems 29, Curran Associates, Inc., pp. 1533–1541.

Bhatia, K., Jain, H., Kar, P., Varma, M., & Jain, P. (2015). Sparse local embeddings for extreme multi-label classification. In: Advances in Neural Information Processing Systems, pp. 730–738.

Bonilla, EV., Chai, KM., & Williams, C. (2008). Multi-task Gaussian process prediction. In: Advances in neural information processing systems, pp. 153–160.

Bui, T. D., Yan, J., & Turner, R. E. (2017). A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *Journal of Machine Learning Research, 18*(104), 1–72.

Csato, L., & Opper, M. (2002). Sparse online Gaussian processes. *Neural Computation, 14,* 641–668.

Dai, Z., Alvarez, M., & Lawrence, N. (2017). Efficient modeling of latent information in supervised learning using Gaussian processes. In: Advances in Neural Information Processing Systems, pp. 5131–5139.

Dezfouli, A., & Bonilla, E.V. (2015). Scalable inference for Gaussian process models with black-box likelihoods. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R (eds) Advances in Neural Information Processing Systems 28, pp. 1414–1422.

Gaure, A., Gupta, A., Verma, V.K., & Rai, P. (2017). A probabilistic framework for zero-shot multi-label learning. In: The Conference on Uncertainty in Artificial Intelligence (UAI), vol 1, p 3

Gibaja, E., & Ventura, S. (2014). Multi-label learning: a review of the state of the art and ongoing research. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 4*(6), 411–444.

Gibaja, E., & Ventura, S. (2015). A tutorial on multilabel learning. *ACM Computer Survey, 47*(3), 52:1–52:38.

He, J., Gu, H., & Wang, Z. (2012). Bayesian multi-instance multi-label learning using Gaussian process prior. *Machine Learning, 88*(1–2), 273–295.

Hensman, J., Matthews, A.G., & Ghahramani, Z. (2015). Scalable variational Gaussian process classification. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics.

Hensman. J., Fusi, N., & Lawrence, N.D. (2013). Gaussian processes for big data. In: Conference on Uncertainty in Artificial Intellegence, auai.org, pp. 282–290.

Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning and Research, 14*(1), 1303–1347.

Jain, V., Modhe, N., & Rai, P. (2017). Scalable generative models for multi-label learning with missing labels. In: Precup D, Teh YW (eds) Proceedings of the 34th International Conference on Machine Learning, PMLR, International Convention Centre, Sydney, Australia, Proceedings of Machine Learning Research, vol 70, pp. 1636–1644

Jain, H., Prabhu, Y., & Varma, M. (2016). Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp. 935–944.

Jasinska, K., Dembczynski, K., Busa-Fekete, R., Pfannschmidt, K., Klerx, T., & Hullermeier, E. (2016). Extreme f-measure maximization using sparse probability estimates. In: International Conference on Machine Learning, pp. 1435–1444.

Kapoor, A., Viswanathan, R., & Jain, P. (2012). Multilabel classification using bayesian compressed sensing. In: Advances in Neural Information Processing Systems, pp. 2645–2653.

Katakis, I., Tsoumakas, G., & Vlahavas, I. (2008). Multilabel text classification for automated tag suggestion. In: Proceedings of the ECML/PKDD, vol 18.

Khandagale, S., Xiao, H., & Babbar, R. (2020). Bonsai: diverse and shallow trees for extreme multi-label classification. Machine Learning pp. 1–21.

Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2007). Ensembles of multi-objective decision trees. In: European conference on machine learning. Springer, pp. 624–631.

Lawrence, ND., Seeger, M., & Herbrich, R. (2002). Fast sparse Gaussian process methods: the informative vector machine. In: Neural Information Processing Systems, 13, MIT Press.

Liu, J., Chang, W.C., Wu, Y., & Yang, Y. (2017). Deep learning for extreme multi-label text classification. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, pp. 115–124.

Lloyd, C., Gunter, T., Osborne, M.A., & Roberts, S.J. (2015). Variational inference for Gaussian process modulated Poisson processes. In: Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, pp. 1814–1822.

Matthews, AG., Hensman, J., Turner, R., & Ghahramani, Z. (2016). On sparse variational methods and the Kullback-Leibler divergence between stochastic processes. In: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, PMLR, Cadiz, Spain, vol 51, pp. 231–239.

McAuley, J., Targett. C., Shi, Q., & Van Den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, pp. 43–52.

Mencia, E.L., & Fürnkranz, J. (2008). Efficient pairwise multilabel classification for large-scale problems in the legal domain. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, pp. 50–65.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In: Burges CJC, Bottou L, Welling M, Ghahramani Z, Weinberger KQ (eds) Advances in Neural Information Processing Systems 26, Curran Associates, Inc., pp. 3111–3119.

Moreno-Muñoz, P., Artés, A., & Álvarez, M. (2018). Heterogeneous multi-output Gaussian process prediction. In: Advances in Neural Information Processing Systems, pp. 6711–6720.

Nam, J., Mencía, E.L., Kim, H.J., & Fürnkranz, J. (2017). Maximizing subset accuracy with recurrent neural networks in multi-label classification. In: Advances in neural information processing systems, pp. 5413–5423.

Niculescu-Mizil, A., & Abbasnejad, E. (2017). Label filters for large scale multilabel classification. In: Artificial Intelligence and Statistics, pp. 1448–1457.

Papanikolaou, Y., & Tsoumakas, G. (2018). Subset labeled LDA: A topic model for extreme multi-label classification. In: International Conference on Big Data Analytics and Knowledge Discovery, Springer, pp. 152–162.

Partalas, I., Kosmopoulos, A., Baskiotis, N., Artieres, T., Paliouras, G., Gaussier, E., Androutsopoulos, I., Amini, M.R., & Galinari, P. (2015). LSHTC: A benchmark for large-scale text classification. arXiv preprint arXiv:150308581

Prabhu, Y., & Varma, M. (2014). Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp. 263–272.

Prabhu, Y., Kag, A., Harsola, S., Agrawal, R., & Varma, M. (2018). Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In: Proceedings of the 2018 World Wide Web Conference, International World Wide Web Conferences Steering Committee, pp. 993–1002

Quiñonero-Candela, J., & Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research, 6,* 1939–1959.

Rasmussen, C. E., & Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. Cambridge: The MIT Press.

Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. *Machine Learning, 85*(3), 333.

Salimbeni, H., Eleftheriadis, S., & Hensman, J. (2018). Natural gradients in practice: Non-conjugate variational inference in Gaussian process models. arXiv preprint arXiv:180309151.

Seeger, M., Williams, C.K.I., & Lawrence, N.D. (2003). Fast forward selection to speed up sparse Gaussian process regression. In: Ninth International Workshop on Artificial Intelligence, MIT Press.

Sheth, R., Wang, Y., & Khardon, R. (2015). Sparse variational inference for generalized GP models. In: Bach F, Blei D (eds) Proceedings of the 32nd International Conference on Machine Learning, PMLR, Lille, France, Proceedings of Machine Learning Research, vol 37, pp. 1302–1311

Si, S., Zhang, H., Keerthi, S.S., Mahajan, D., Dhillon, I.S., & Hsieh, C.J. (2017). Gradient boosted decision trees for high dimensional sparse output. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, pp. 3182–3190

Siblini, W., Kuntz, P., & Meyer, F. (2018). Craftml, an efficient clustering-based random forest for extreme multi-label learning.

Snelson, E., & Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In: Weiss Y, Schölkopf B, Platt JC (eds) Advances in Neural Information Processing Systems 18, pp. 1257–1264.

Stoyan, D. (1996). Hans wackernagel: Multivariate geostatistics. An introduction with applications. with 75 figures and 5 tables. springer-verlag, berlin, heidelberg, new york, 235 pp., 1995, dm 68.-isbn 3-540-60127-9. Biometrical Journal 38(4):454–454

Tagami, Y. (2017). Annexml: Approximate nearest neighbor search for extreme multi-label classification. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp. 455–464.

Teh, Y.W., Seeger, M., & Michael, J. (2005). Semiparametric latent factor models. In: Workshop on Artificial Intelligence and Statistics 10.

Titsias, M.K. (2009). Variational learning of inducing variables in sparse Gaussian processes. In: International Conference on Artificial Intelligence and Statistics, pp. 567–574

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2008). Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08), sn, vol 21, pp. 53–59.

Tsoumakas, G., & Katakis, I. (2007). Multi label classification: an overview. *International Journal of Data Warehouse and Mining, 3*(3), 1–13.

Weston, J., Bengio, S., & Usunier, N. (2011). Wsabie: Scaling up to large vocabulary image annotation. In: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI.

Wetzker, R., Zimmermann, C., & Bauckhage, C. (2008). Analyzing social bookmarking systems: A del. icio. us cookbook. In: Proceedings of the ECAI 2008 Mining Social Data Workshop, pp. 26–30.

Wydmuch, M., Jasinska, K., Kuznetsov, M., Busa-Fekete, R., & Dembczynski, K. (2018). A no-regret generalization of hierarchical softmax to extreme multi-label classification. In: Advances in Neural Information Processing Systems, pp. 6355–6366.

Yen, I.E., Huang, X., Dai, W., Ravikumar, P., Dhillon, I., & Xing, E. (2017). Ppdsparse: A parallel primal-dual sparse method for extreme classification. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp. 545–553.

Yen, I.E.H., Huang, X., Ravikumar, P., Zhong, K., & Dhillon, I. (2016). Pd-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In: International Conference on Machine Learning, pp 3069–3077.

You, R., Zhang, Z., Wang, Z., Dai, S., Mamitsuka, H., & Zhu, S. (2019). Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In: Advances in Neural Information Processing Systems, pp. 5820–5830.

Yu, H.F., Jain, P., Kar, P., & Dhillon, I. (2014). Large-scale multi-label learning with missing labels. In: International Conference on Machine Learning, pp. 593–601.

Zhang, M. L., & Zhou, Z. H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition, 40*(7), 2038–2048.

Zhang, M., & Zhou, Z. (2013). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*. https://doi.org/10.1109/TKDE.2013.39.

Zubiaga, A. (2012). Enhancing navigation on wikipedia with social tags. arXiv preprint arXiv:12025469.