# Performance Prediction via Bayesian Matrix Factorisation for Multilingual Natural Language Processing Tasks

**Viktoria Schram  Daniel Beck  Trevor Cohn**
School of Computing and Information Systems
The University of Melbourne
vschram@student.unimelb.edu.au
{d.beck,t.cohn}@unimelb.edu.au

## Abstract

Performance prediction for Natural Language Processing (NLP) seeks to reduce the experimental burden resulting from the myriad of different evaluation scenarios, e.g., the combination of languages used in multilingual transfer. In this work, we explore the framework of Bayesian matrix factorisation for performance prediction, as many experimental settings in NLP can be naturally represented in matrix format. Our approach outperforms the state-of-the-art in several NLP benchmarks, including machine translation and cross-lingual entity linking. Furthermore, it also avoids hyperparameter tuning and is able to provide uncertainty estimates over predictions.

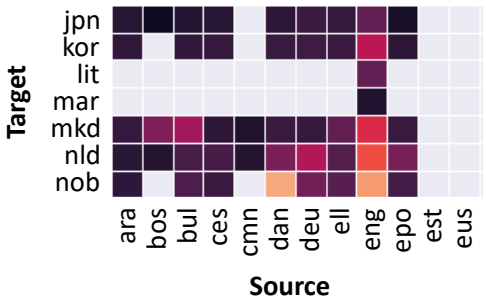


Figure 1: Matrix representation showing a subset of the performance scores for Wiki MT (Schwenk et al., 2021). Black: Lowest performance scores. Light orange: Highest performance scores. Grey: No performance score available. The goal is to predict the performance of language pairs depicted by grey cells.

## 1 Introduction

Natural language processing (NLP) is an empirical discipline, with progress driven by a myriad of tasks, domains, computational models and datasets (Xia et al., 2020). Models are often developed to be applicable to a large number of languages. Given that there are more than 7000 languages spoken in the world (Haddow et al., 2022), evaluating the model performance of each possible NLP scenario leads to a combinatorial explosion and would require excessive time for training and testing, as well as the significant costs of evaluation resource creation. This is very time consuming and computationally expensive, especially for large models and massively multilingual applications (Xia et al., 2020; Sharir et al., 2020).

To illustrate, for neural machine translation (MT) models, retraining and testing for each new language pair is required (Fan et al., 2021), which quickly increases the number of experiments (as shown in Figure 1). In those scenarios, whether the model is appropriate for the considered test languages is not clear upfront. Similarly, for cross-lingual transfer (TSF) tasks like parsing (Das and Sarkar, 2020) or part-of-speech (POS) tagging (de Vries et al., 2022), the performance on all languages of interest is only known after training and testing has been performed. This is particularly interesting for the development of NLP technologies for low-resource languages (de Vries et al., 2022).

In this paper we propose a method for performance prediction which can solve the above problems through automatically estimating performance scores of machine learning models, thus avoiding the need for excessive training and test cycles, and data creation costs. The goal is to predict the performance of a model solely based on past experimental records – a task of great interest for decision making in aforementioned high-cost applications. We propose to use Bayesian probabilistic matrix factorization methods and incorporate context features describing the language pairs of interest. This is realisable because many performance prediction problems in NLP can be modelled as a matrix, as illustrated in Figure 1. The result will inevitable be an incomplete matrix because some data are missing, as they were either to costly to obtain or no datasets were available (Xia et al., 2020; Schwenk et al., 2021).



*Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1790–1801
May 2-6, 2023 

Our key contribution is a Bayesian approach for performance prediction, contrasting with prior work which is exclusively frequentist. This confers the following advantages: 1) it enables us to quantify the uncertainty of our predictions by providing credible intervals (CIs); and 2) predictions of performance scores can be provided in a more principled way via a posterior distribution after integrating over the model's parameters and hyperparameters. We provide an extensive experimental study of matrix factorization methods, including our proposed Bayesian technique, applied to several important benchmarks, and show several improvements over the state-of-the-art.

## 2 Problem Formulation and Methods

As demonstrated in Figure 1, the performance scores for a suitable NLP task can be arranged in a matrix with the rows displaying the source languages and the columns showing the target languages[1]. The matrix is the score matrix $\mathbf{R} \in \mathbb{R}^{N_s \times N_t}$ and each cell $(s, t)$ contains either the performance score for a source-to-target language relationship or is empty. The number of considered source and target languages is denoted by $N_s$ and $N_t$, respectively.

The performance of a model, measured by an evaluation metric (e.g. BLEU, Papineni et al. (2002)), depends on properties like the model architecture $\mathcal{M}$, training dataset $\mathcal{D}$, languages $\mathcal{L}$, the training procedure $\mathcal{P}$ and the test dataset $\mathcal{D}'$. The score matrix $\mathbf{R}$ is then given by

$$\hat{\mathbf{R}}_{\mathcal{L},\mathcal{D}} = f_\theta([\Phi_\mathcal{L}; \Phi_\mathcal{D}]),$$

with $f_\theta(\cdot)$ being a function which denotes a linear or non-linear relationship of its arguments and $\Phi_\mathcal{X}$ represents the feature set of each characteristic $\mathcal{X}$. The dependency on test set features is omitted. We assume that the distributions of training and test datasets are the same, ignoring a possible domain shift (following Xia et al. (2020)).

As the completion of an arbitrary matrix is ill-posed, we reformulate $f_\theta(\cdot)$ as a low-rank matrix approximation problem. In its simplest, noiseless form, the matrix $\mathbf{R}$ can be reconstructed as (Koren et al., 2009)

$$\mathbf{R} \approx \mathbf{W}^T \mathbf{H}, \quad (1)$$

[1]This is an assumption, appropriate for our considered tasks. Without any loss of generality, this scenario can be expanded to other NLP settings which can be reformulated as a matrix, e.g. hyperparameter search for language models.

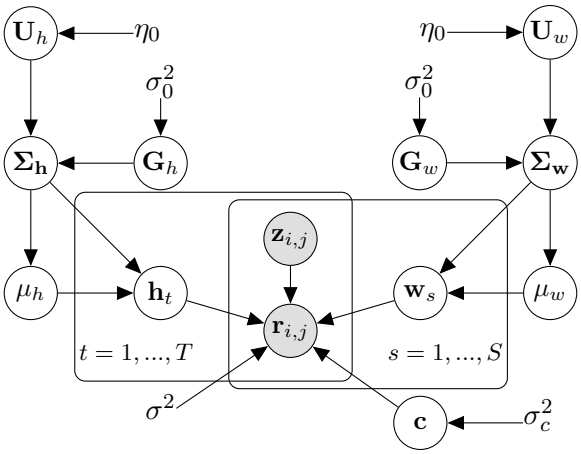


Figure 2: The graphical model for our approach based on Bayesian Probabilistic Matrix Factorisation with context information.

using two lower rank matrices $\mathbf{W} \in \mathbb{R}^{D \times N_s}$ and $\mathbf{H} \in \mathbb{R}^{D \times N_t}$, with $\mathbf{W}$ and $\mathbf{H}$ being the latent matrices for the source and target languages, respectively. Due to the low-rank assumption, for the latent dimensions it holds that $D \ll N_s, N_t$. Hence, each individual score $r_{s,t}$ of a source-target pair can be calculated by $r_{s,t} \approx \mathbf{w}_s^T \mathbf{h}_t$, where each latent source and target language vector is given by a low-dimensional vector, $\mathbf{w}_s^T \in \mathbb{R}^D$, $\mathbf{h}_t \in \mathbb{R}^D$, respectively (Cabral et al., 2013; Chen et al., 2018b).

The matrix factorisation (MF) approach described above is inspired by research on recommender systems, which use latent factor models for preference predictions (Koren et al., 2009). In those models, the low rank behavior is a widely used assumption. We assume in our model that the performance of a bilingual task is determined by a small number of unobserved factors. Using a linear factor model the performance for translating from a source language is modeled by linearly combining target language factor vectors using source language specific coefficients.

### 2.1 Bayesian Probabilistic Matrix Factorisation

Our approach is represented by the graphical model shown in Figure 2. It illustrates a linear probabilistic model, where the conditional distribution over the observed ratings, assuming Gaussian observation noise, is given as

$$p(\mathbf{R}|\mathbf{W}, \mathbf{H}, \mathbf{c}, \mathbf{z}, \sigma^2) = \prod_{s=1}^{S} \prod_{t=1}^{T} [\mathcal{N}(r_{s,t}|\mathbf{w}_s^T \mathbf{h}_t + \sum_{i=1}^{N} c_i z_{s,t}^i, \sigma^2)]^{I_{s,t}}, \quad (2)$$

with $I_{s,t}$ being the indicator function that is 1 if a source language $s$ has a performance score for a target language $j$ and 0 otherwise. $\mathcal{N}(\mu, \sigma^2)$ denotes a Gaussian distribution with mean $\mu$ and variance $\sigma^2$. The mean of this likelihood function consists of a latent factor model $\mathbf{w}_s^T \mathbf{h}_t$ and is also additionally regressed on the available context information given by $z_{s,t}^i$ for $i \in 1, ..., N$ available context features (Chen et al., 2018b). The prior for the regression coefficients $c_i$ of each context feature $z_i$ is given by $p(c_i|\sigma_c^2) = \mathcal{N}(c_i|0, \sigma_c^2)$. Furthermore, spherical Gaussian priors are placed for source and target language latent feature vectors given by

$$p(\mathbf{W}|\sigma_w^2) = \prod_{s=1}^{S} \mathcal{N}(w_s|\mu_w, \mathbf{\Sigma}_w) \quad (3)$$

$$p(\mathbf{H}|\sigma_h^2) = \prod_{t=1}^{T} \mathcal{N}(h_t|\mu_h, \mathbf{\Sigma}_h), \quad (4)$$

with hyperparameters $\Theta_w = \{\mu_w, \mathbf{\Sigma}_w\}$ and $\Theta_h = \{\mu_h, \mathbf{\Sigma}_h\}$, where $\mathbf{\Sigma}$ is the covariance matrix. In contrast to the original work (Salakhutdinov and Mnih, 2008) on which we base our approach, we choose to incorporate the Lewandowski-Kurowicka-Joe (LKJ) prior (Lewandowski et al., 2009) into the computation of samples for the covariance matrices. This method is better suited for modern Bayesian computations, due to sampling difficulties the inverse-Wishart prior causes and its restrictive form (Team, 2018; Barnard et al., 2000). A correlation matrix $\mathbf{U}$, with elements $u_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$ and $\sigma$ being the standard deviation, can be reformulated into a covariance matrix $\mathbf{\Sigma}$ using the separation strategy (Barnard et al., 2000) as

$$\mathbf{\Sigma} = \mathbf{GUG}, \quad (5)$$

with $\mathbf{G} = \sqrt{\text{diag}(\mathbf{\Sigma})}$. The matrix $\mathbf{U}$ is sampled from the LKJ prior given by

$$p(\mathbf{U}|\eta_0) = c \cdot \det(\mathbf{U})^{\eta_0 - 1},$$

indicating a uniform distribution on correlation matrices for $\eta_0 = 1$, a normalization constant $c$ and $\det(\cdot)$ denoting the determinant. As a prior for $\mathbf{G}$ we choose the halfnormal distribution (Seyboldt, 2019), as it holds that $\sigma > 0$, which is given for each element $g \in \mathbf{G}$ by

$$p(g|\sigma_0^2) = \frac{\sqrt{2}}{\sigma_0 \sqrt{\pi}} \exp\left(-\frac{g^2}{2\sigma_0^2}\right), \; g \geq 0,$$

with unit variance $\sigma_0^2$. This is equivalent to placing an inverse-Gamma distribution on the precision. The covariance matrix $\mathbf{\Sigma}$ obtained according to Equation (5) is used to parameterise the priors given in Equations (3) and (4) and the Gaussian prior of their means. More information can be found in Salakhutdinov and Mnih (2008).

The predictive distribution is obtained by marginalizing over the model's parameters and hyperparameters. Since exact evaluation of this predictive distribution is analytically intractable, we use approximate inference via Markov chain Monte Carlo (MCMC) sampling, leading to an approximation of the predictive posterior as (Salakhutdinov and Mnih, 2008):

$$p(\mathbf{R}^*|\mathbf{R}, \Theta_0) \approx \frac{1}{K} \sum_{k=1}^{K} p(\mathbf{R}^*|\mathbf{W}^k, \mathbf{H}^k), \quad (6)$$

for $\Theta_0 = \{\eta_0, \sigma_0^2\}$ and $\mathbf{R}^*$ denoting the matrix of predictions on the test set. The $K$ samples are generated by running a Markov chain. Its stationary distribution will be the posterior distribution over the model parameters and hyperparameters $\{\mathbf{W}, \mathbf{H}, \Theta_w, \Theta_h\}$.

### 2.2 Credible Intervals

One measure of uncertainty in Bayesian inference is the credible interval (CI) or more generally a credible set (Casella and Berger, 2021). A $100(1-\beta)\,\%$ equal tail CI for a random variable $X$ is an interval $[a, b]$ such that the probability that $X$ lies in the interval is $1 - \beta$, given as

$$P(X \in [a, b]) = 1 - \beta.$$

In Bayesian inference, analyses are made considering the posterior distribution of the parameter of interest. Hence, a CI provides us with upper and lower bounds which define an interval with the probability of interest around the mean of the posterior distribution (Hespanhol et al., 2019; Rice and Ye, 2022). It quantifies how precise the obtained posterior, i.e. the posterior belief, is, with a narrower interval around the point estimate indicating more certainty in the predictions. Hence, using the samples from Equation (6), an approximation of the posterior predictive distributions for each language pair can be obtained and used to construct CIs in addition to the point estimates. This is useful for performance prediction in NLP settings, as the widths and bounds of the obtained CIs can guide further decision-making regarding whether a model is theoretically worth deploying.

| Task | dim | cells | empty | test | feats |
|---|---|---|---|---|---|
| $\text{MT}^{\text{TSF}}$ | $54 \times 54$ | 2916 | 0 | 573 | 21 |
| $\text{POS}^{\text{TSF}}$ | $26 \times 60$ | 1560 | 14 | 29 | 15 |
| $\text{MT}^{\text{Wiki}}$ | $39 \times 39$ | 1521 | 526 | 199 | 22 |
| $\text{Parsing}^{\text{TSF}}$ | $30 \times 30$ | 900 | 0 | 174 | 15 |
| $\text{EL}^{\text{TSF}}$ | $9 \times 54$ | 486 | 0 | 96 | 12 |
| BLI | $15 \times 15$ | 225 | 95 | 26 | 6 |

Table 1: Matrix specifications for each task. Dim: Matrix dimensions. Cells: Number of cells in each matrix. Empty: Number of empty cells in the matrix, not being cells on the diagonal. Test: Number of cells used in the test set. Feats: Number of features.

### 2.3 Non-Bayesian Matrix Factorisation Methods

Probabilistic MF (PMF) has been shown to perform well on sparse and imbalanced datasets (Mnih and Salakhutdinov, 2007). The conditional distribution over the observed ratings, assuming Gaussian observation noise given by Equation (2), without considering additional context information. As in Bayesian PMF (BPMF) spherical Gaussian priors are placed on $\mathbf{W}$ and $\mathbf{H}$ following Equations (3) and (4), assuming $\mu_w = \mu_h = 0$; $\mathbf{\Sigma}_w = \sigma_w^2\mathbf{I}$ and $\mathbf{\Sigma}_h = \sigma_h^2\mathbf{I}$. Learning the latent source and target matrices is done via maximizing the log-posterior, where all hyperparameters, being the observation noise and prior variances, are fixed. This is equal to the following optimization problem, where latent factor representations are learned by finding the matrices $\mathbf{W}$ and $\mathbf{H}$ which minimize the regularized squared error, given by (Chen et al., 2018b)

$$\mathbf{W}^*, \mathbf{H}^* = \arg\min_{\mathbf{W},\mathbf{H}} \frac{1}{2} \sum_{(s,t)\in\delta(\mathbf{R})} (\mathbf{r}_{s,t} - \mathbf{w}_s^T \mathbf{h}_t)^2 + \beta_w \sum_{s=1}^{N_s} ||\mathbf{w}_s||_2^2 + \beta_h \sum_{t=1}^{N_t} ||\mathbf{h}_t||_2^2, \quad (7)$$

where $||\cdot||_2^2$ is the squared $l_2$ norm[2]. Additionally, separate regularization parameters for $\mathbf{W}$ and $\mathbf{H}$ are assumed, being $\beta_w$ and $\beta_h$, respectively. In the case of missing values, the optimization problem is on the set of known ratings $\delta(\mathbf{R})$.

Additionally, the framework of MF offers the option of biased MF, which provides a means of incorporating context. It incorporates model characteristics of each source language, $b_s$, and target language $b_t$ and considers a global tendency $\mu$ which is independent of source-target language interactions. The biases account for the fact that certain performance values might contain universal shifts or exhibit systematic tendencies with respect to certain source and target languages. Furthermore, it is common that all performance predictions are non-negative and in a certain range, i.e., the global bias $\mu$ accounts for global effects. The predictions are given by (Chen et al., 2018b)

$$\hat{r}_{s,t}^{\text{CTX}} = \mathbf{w}_s^{\text{T}} \mathbf{h}_t + \sum_{i=1}^{N} c_i z_{s,t}^i + \mu + b_s + b_t. \quad (8)$$

During learning, the optimization problem in Equation (7) has to be adapted to $\hat{r}_{s,t}^{\text{CTX}}$ accordingly, also adding additional regularization terms for $b_s$ and $b_t$.

The non-probabilistic approaches, but also PMF need manual control of their hyperparameters, which makes those approaches computationally expensive to develop compared to BPMF.

## 3 Experiments and Analysis

Following the evaluation setting of *NLPerf* (Xia et al., 2020), we consider a set of tasks described in the following, all of which allow a reformulation into a matrix representation. The performance scores are predicted for bilingual lexicon induction (BLI); machine translation on aligned Wikipedia data ($\text{Wiki}^{\text{MT}}$), and with cross-lingual transfer for translation into English ($\text{MT}^{\text{TSF}}$); cross lingual dependency parsing ($\text{Parsing}^{\text{TSF}}$); cross-lingual POS tagging ($\text{POS}^{\text{TSF}}$) and cross-lingual entity linking ($\text{EL}^{\text{TSF}}$). Despite also considering cross-lingual transfer tasks, for reasons of simplicity and without any loss of generality, we refer to each cell in the matrix as a *source-target* language pair unless otherwise stated. All tasks and their matrix properties are stated in Table 1. More information about the models underlying each task can be found in Appendix A.

To allow a fair comparison to the closest related work *NLPerf* (Xia et al., 2020), we consider an identical set of features for our predictions: Dataset size, word/subword vocabulary size, average sentence length, word/subword overlap, type-token ratio, type-token ratio distance, single tag type, fused tag type, average tag length per word, dependency arcs matching WALS features and six distance features from the URIEL Typological Database (Littell

[2]Solving this optimization problem to obtain a point estimate is referred to as MF in our experiments. We use a Bayesian approach for PMF and therefore apply MCMC sampling, which is another alternative and can be easily extended via hyperpriors to BPMF.

| Task | $\text{MT}^{\text{Wiki}}$ | $\text{Parsing}^{\text{TSF}}$ | $\text{EL}^{\text{TSF}}$ | $\text{POS}^{\text{TSF}}$ | $\text{MT}^{\text{TSF}}$ | BLI |
|---|---|---|---|---|---|---|
| NLPerf | 2.49 | 6.23 | 7.44 | 7.44 | 1.42 | **8.78** |
| MF $D = 10$ | $2.74 \pm 0.08$ | $3.43 \pm 0.16$ | $6.47 \pm 0.26$ | $6.81 \pm 0.19$ | $1.42 \pm 0.02$ | $11.24 \pm 0.45$ |
| MF $D = 20$ | $2.73 \pm 0.07$ | $3.36 \pm 0.08$ | $6.33 \pm 0.25$ | $6.03 \pm 0.10$ | $1.40 \pm 0.02$ | $12.05 \pm 0.72$ |
| MF CTX $D = 10$ | $2.19 \pm 0.07$ | $2.37 \pm 0.04$ | $5.45 \pm 0.19$ | $3.98 \pm 0.07$ | $1.42 \pm 0.02$ | $10.92 \pm 0.66$ |
| MF CTX $D = 20$ | $\mathbf{2.15 \pm 0.05}$ | $\mathbf{2.25 \pm 0.05}$ | $5.33 \pm 0.15$ | $\mathbf{3.86 \pm 0.07}$ | $\mathbf{1.19 \pm 0.02}$ | $10.80 \pm 0.61$ |
| PMF $D = 10$ | $2.88 \pm 0.05$ | $3.27 \pm 0.02$ | $11.19 \pm 0.08$ | $5.87 \pm 0.03$ | $1.44 \pm 0.00$ | $11.65 \pm 0.37$ |
| PMF $D = 20$ | $2.92 \pm 0.04$ | $3.34 \pm 0.00$ | $11.25 \pm 0.08$ | $5.78 \pm 0.05$ | $1.43 \pm 0.01$ | $11.83 \pm 0.41$ |
| BPMF $D = 10$ | $2.64 \pm 0.03$ | $3.77 \pm 0.00$ | $5.00 \pm 0.06$ | $23.03 \pm 3.33$ | $1.61 \pm 0.03$ | $9.23 \pm 0.11$ |
| BPMF $D = 20$ | $2.63 \pm 0.03$ | $2.99 \pm 0.04$ | $5.00 \pm 0.09$ | $6.44 \pm 0.13$ | $1.65 \pm 0.07$ | $9.30 \pm 0.05$ |
| BPMF $D = 30$ | $2.59 \pm 0.06$ | $2.96 \pm 0.03$ | $4.98 \pm 0.05$ | $6.21 \pm 0.1$ | $1.64 \pm 0.03$ | $9.07 \pm 0.04$ |
| BPMF CTX $D = 10$ | $2.42 \pm 0.00$ | $3.69 \pm 0.06$ | $4.97 \pm 0.05$ | $17.56 \pm 1.77$ | $1.53 \pm 0.02$ | $9.18 \pm 0.20$ |
| BPMF CTX $D = 20$ | $2.35 \pm 0.11$ | $2.92 \pm 0.05$ | $\mathbf{4.87 \pm 0.02}$ | $6.17 \pm 0.10$ | $1.57 \pm 0.04$ | $9.20 \pm 0.07$ |
| BPMF CTX $D = 30$ | $2.51 \pm 0.02$ | $2.96 \pm 0.03$ | $5.15 \pm 0.19$ | $6.02 \pm 0.10$ | $1.53 \pm 0.02$ | $9.22 \pm 0.04$ |

Table 2: RMSE $\pm$ standard deviation for various all tasks considered. $D$: Latent factor dimension. CTX: Approaches incorporating context features.

et al., 2017). Further information regarding each feature can be found in Xia et al. (2020).

All features were normalized using their corresponding z-score. For each task, (except BLI), we predict scores for a single model. This means that there is exactly one underlying statistical or neural model considered for each task. For BLI however, our dataset consists of two metrics *Vecmap* and *Muse*. The performance is obtained as in the equivalent multi-model scenario in Xia et al. (2020), by averaging over both metrics.

The *NLPerf* framework is based on gradient boosting trees (Friedman, 2001) and implemented with XGBoost (Chen and Guestrin, 2016). The parameterisation of this predictor was kept as in (Xia et al., 2020), assuming a squared error as the objective function for the regression, a fixed learning rate of 0.1, a maximum tree depth of 10 and the number of trees being 100. The default setting was used for the regularization terms. We evaluate the performance of our methods by using the root mean squared error (RMSE).

Predictions for MF are obtained according to Equation (8) but through omitting the context information, whereas MF CTX makes use of the latter. The conditional distribution over observed ratings as given in Equation (2) is used to obtain predictions for BPMF, and adapted accordingly for PMF. All non-Bayesian models are trained using nested 5-fold cross validation and stochastic gradient descent (SGD) over 2000 training iterations. Referring to Figure 1, note that during training of the latent vectors $\mathbf{w}_s$ and $\mathbf{h}_t$, all available scores in the corresponding column $s$ and row $t$ are used. The regularization parameters are chosen using a validation set. All non-probabilistic experiments are averaged over 10 runs. Training of the probabilistic models was performed using MCMC sampling and the results are averaged over 2 runs, due to MCMC time complexity.[3] More information about the execution times of each experiment is given in Appendix B.

### 3.1 Random Train-Test Splits

We start by investigating the behaviour of all methods on a randomly chosen training and test split containing a subset of language pairs. Note that we create one split per task and keep it constant across all methods to allow fair comparison of the obtained results. An example for such a train-test split is depicted in Figure 3 on the left for BLI.

The RMSE and standard deviations of our predicted performances for all tasks are provided in Table 2. For all single model tasks, the MF approaches consistently outperform the state-of-the-art *NLPerf* (Xia et al., 2020). While general MF already outperforms *NLPerf* in all but one setting, using context information (MF CTX) leads to further improvements in terms of RMSE – outperforming the SOTA with reductions in RMSE of up to 64% ($\text{Parsing}^{\text{TSF}}$). While the improvement is significant for tasks like $\text{EL}^{\text{TSF}}$, $\text{POS}^{\text{TSF}}$ and the aforementioned $\text{Parsing}^{\text{TSF}}$, we observe a smaller but still noticeable reduction in RMSE for $\text{Wiki}^{\text{MT}}$ and $\text{MT}^{\text{TSF}}$ of around 13% and 16%, respectively. Note however, that the RMSE scores cannot be directly compared across tasks as the scales of all

[3] Code is available at https://github.com/vschram/PP-via-Bayesian-MF-for-Multilingual-NLP under the CCBY-SA 4.0 license

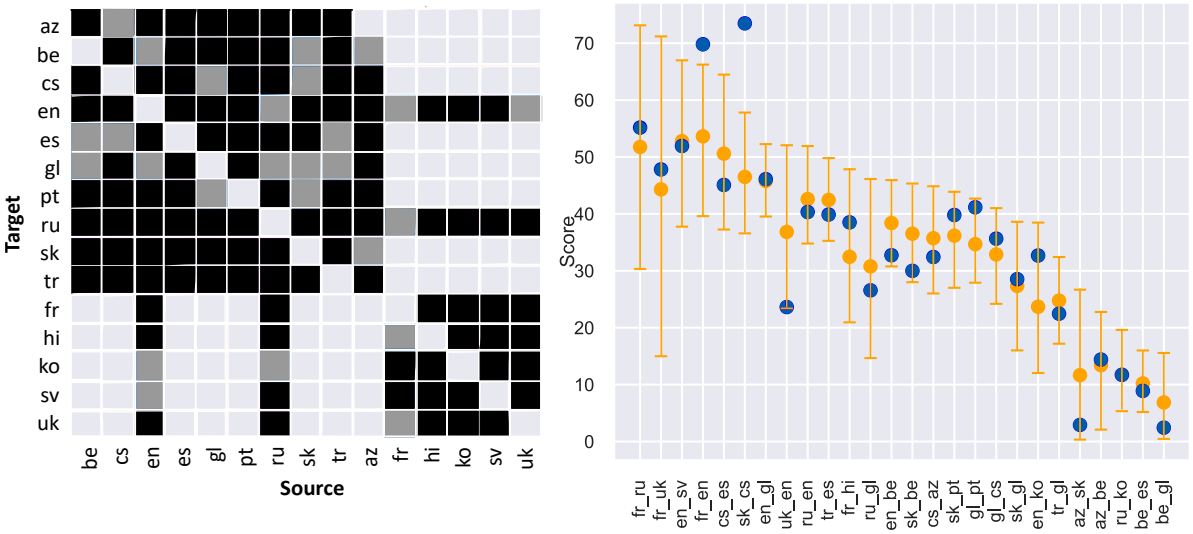


Figure 3: **Left:** One train-test split of the available BLI data set in matrix format. Light grey: Unavailable data. Dark grey: Test data. Black: Training data. **Right:** 95 % Credible intervals (CI) for the predicted scores for the scenario on the left. Orange: Predicted scores. Blue: Actual scores. Method BPMF, dimensionality of latent vectors: 10.

evaluation metrics differ.

While the Bayesian version BPMF CTX also outperforms the *NLPerf* baseline in all but one single model task, we observe particularly good results for the task $\text{EL}^{\text{TSF}}$, where BPMF CTX outperforms all other methods and achieves new state-of-the-art results with a relative reduction in RMSE of 35%. As introduced in Section 2, the non-Bayesian PMF requires hyperparameter optimization and as shown in Table 2 for all tasks but $\text{MT}^{\text{TSF}}$, BPMF outperforms PMF – potentially due to non-optimal hyperparameter choices for PMF. This might also be the cause for the PMF results in the $\text{EL}^{\text{TSF}}$ experiments, while they converge to a solution, the obtained model does not represent the underlying data well. The outliers for $\text{POS}^{\text{TSF}}$ are caused by divergences, which can be avoided by using a higher number of latent dimensions. Further research is required in those cases to understand how the models in those cases can be better adapted to the data and is part of future work.

Observing all single model results obtained, we conclude that if a dataset is given for which a matrix can be constructed with a limited number of missing cells, MF approaches provide a strong alternative to gradient boosting trees. Results for all tasks show that linear MF approaches with rather small latent dimension sizes of 10 or 20 show often significant performance improvements over *NLPerf*.

In contrast to the single model tasks, the BPMF approaches achieve competitive performance in the BLI two model scenario, but none of the MF approaches are able to outperform the SOTA. We suspect this might be due to the matrix size and available data, which is significantly smaller than in all other tasks (Table 1). Furthermore, only 6 distance features are available for this setting, indicating that additional features like *dataset size* and *vocabulary size* (among others) that are available for the other tasks might play a important role in facilitating accurate performance prediction.

### 3.2 Uncertainty of the Predicted Scores

The Bayesian approaches offer the advantage of providing a measure of uncertainty in terms of CIs. An example showing the 95 % CIs for all language-pairs available in the test set is depicted in Figure 3. Predicted performance scores for cells where only limited training data in the corresponding rows and columns is available show wide approximations of the predictive posterior distribution in terms of CIs, e.g. the language pairs “fr_uk” or “fr_hi”. In contrast, those with more training data lead to narrower CIs like for “be_es” or “en_be”.

Apart from having a direct measure of uncertainty, one can use upper confidence bounds (UCB) and lower confidence bounds (LCB) of the obtained CIs to guide further decision-making (Auer, 2002). Maximum UCBs can help to determine language-pairs for which the underlying model could be especially high-performing, justifying the cost to

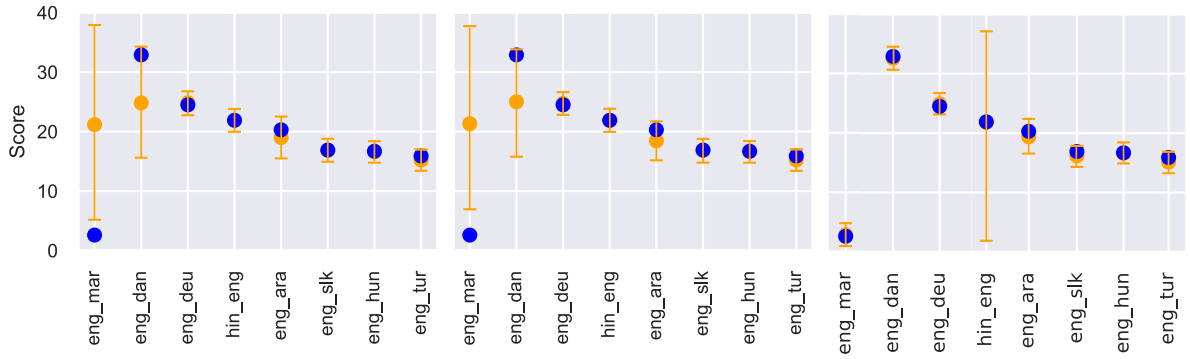


Figure 4: Credible intervals for the task: $MT^{Wiki}$. Orange: Predicted Score. Blue: Actual scores. **Left:** BPMF, latent factor dimensionality: 10. **Middle:** BPMF, latent factor dimensionality: 20. **Right:** BPMF CTX, latent factor dimensionality: 20. An increase in latent factor dimensionality and adding additional features leads to narrower CIs. However, adding features to the language pair "hin_eng" causes a wider CI, indicating that the used features might be unsuitable for this language-pair.

train and acquire the necessary datasets for such language-pairs. The example given in Figure 3 on the right shows CIs for the BLI task. Using the UCBs, the language-pairs "fr_ru", "fr_uk" and "en_sv" could potentially provide us with the highest achievable scores (between 65 and 75). In contrast, using the minimum UCBs, we can conclude that only very low possible maximum scores for the language pairs "ru_ko", "be_es" and "be_gl" can be achieved. Those will be between 15 and 20, therefore it is discouraged to train and test the underlying models on those. Moreover, the decision whether a model is suited for a language-pair can also be based on the maximum LCBs, indicating the potential that a performance score will be higher or equal then the obtained bounds. For the considered task, this is true for the language pairs "fr_en", "en_gl" and "en_sv".

Furthermore, CIs provide important additional insights that help to interpret design choices. Figure 4 shows the impact and importance that an increase in latent factor dimensionality as well as the addition of additional features has on the performance of the task $MT^{Wiki}$. The left figure shows predictions and corresponding CIs using BPMF for a latent vector dimension of 10. After increasing the dimensionality to 20, the resulting CIs get slightly narrower as shown in the middle figure. Finally on the right, it can be observed that after adding CTX for language pairs like "eng_dan" and "eng_hun", the CIs get significantly narrower and indicate increased certainty of the predictions. However, the CI for "hin_eng" interestingly becomes wider, indicating that the additionally chosen features for this language pair might be less beneficial and even hurt performance prediction. Further investigation of the feature set shows that "hin_eng" exhibits the highest value of the feature *average sentence length of the source language*, leading to the conclusion that extreme feature values lead to more uncertainty in the predictions, which is to be expected as the Bayesian model has not encountered this feature value during training.

| | LOLO transfer | LOLO target |
|---|---|---|
| NLPerf | **8.08** | 10.62 |
| MF | $13.29 \pm 0.01$ | $11.86 \pm 0.02$ |
| MF CTX | $11.16 \pm 0.02$ | $\mathbf{6.78 \pm 0.02}$ |
| Bayes | $13.09 \pm 0.00$ | $10.85 \pm 0.00$ |
| Bayes CTX | $13.04 \pm 0.00$ | $10.68 \pm 0.00$ |

Table 3: RMSE $\pm$ standard deviation for $Parsing^{TSF}$. Experiments leaving-one-language-out (LOLO). Dimensionality of latent factors: 20.

It is clear from Figures 3 and 4 that the credible intervals are imperfect, not always including the test results. This is to be expected 5% of the time by chance, following the definition of the credible interval. However, deviations can arise through modelling biases. Consider the language pairs "fr-en" and "sk-cs" in Figure 3. The scores for these instances are not seen during training, and thus generalisation errors may occur. For the former, there is lack of training data for a translation from French to other languages. For the latter, although both languages are closely related, there is not enough training data support confident predictions. Further-

more, BPMF does not use any additional features. As visible from Figure 4 for the language pair "engmar", adding context features provides additional information and greatly improves the accuracy of the Bayesian credible interval.

| | |
|---|---|
| NLPerf | 8.78 |
| LR | 17.26 |
| BPMF CTX $D = 20$ | $9.20 \pm 0.07$ |

Table 4: Contrasting the performance of predictors, being NLPerf, linear regression (LR) and BPMF CTX.

### 3.3 Leave-one-Language-out Scenarios

We further investigate the importance of additional features for performance prediction in cold start scenarios. In such settings, a new language occurs in the test set which has not been encountered during training. This is referred as leave-one-language-out (LOLO). We conducted the experiment for Parsing$^{\text{TSF}}$ as this includes a complete matrix with the same target and transfer languages as rows and columns. This provides us with the opportunity to contrast the performance of the predictors regarding having the same target and transfer languages.

The vectors of the latent factor models of MF and MF CTX are initialized to small random values during training. Thereby, the latent vector corresponding to the missing language will not be further changed as there is no training data available. Note that MF leverages additionally the information of additional bias as shown in Equation (8). In the Bayesian setting, the latent vectors are sampled according to the probabilistic model from the specified priors. Further training details are kept the same as in Section 3.1.

The results are shown in Table 3. Our experiments demonstrate that when leaving one target language out, the predicted performance of MF CTX significantly outperforms all other approaches. However, when leaving one transfer language out, although the performance scores predicted by MF CTX are better then other MF approaches, they do not outperform *NLPerf*. These results indicate that the performance of a predictor based on the MF framework seems to heavily depend on the missing LOLO direction, being target or transfer – rendering MF CTX well suited for cold start predictions when performance scores for unseen target languages are of interest.

### 3.4 On the Effects of Non-Linearity

In Table 4 the performance in terms of RMSE of XGBoost, linear regression (LR) and BPMF CTX for BLI is shown. The proposed MF methods were not able to outperform the SOTA in this scenario. However, while *NLPerf* is a non-linear method, our approaches are all bilinear. Comparing to the performance of LR, BPMF CTX clearly outperforms, having the additional advantage of being able to provide measures of uncertainty. This suggests that further investigations are necessary in whether a non-linear model would be better suited for the prediction of performance scores for BLI.

## 4 Related Work

Matrix completion (MC) is an important technique used to recover a matrix from a subset of its entries, widely applied and studied in many areas of research like machine learning, data science, signal processing and communications (Du and Swamy, 2013; Yuchi et al., 2022). In NLP, it is found in keyword searches or recommender systems, among others (Chen et al., 2018b,a). MC in the context of recommender systems can be interpreted as e.g. the task of predicting a product to recommend to costumers. In the famous *Netflix price competition*, targeting recommender systems for movies, it has been shown that an approach based on matrix factorization (MF) via low-rank latent factor models is the basis for the best algorithm to predict user ratings (Koren et al., 2009; Chen et al., 2018b).

There are two main directions of performance prediction in machine learning. The performance can either be predicted as a function of certain training dataset properties or as a function of its training time or number of iterations (Kolachina et al., 2012). Following the former, current state of the art approaches like *NLPerf* (Xia et al., 2020) or *LITMUS* (Srinivasan et al., 2022) leverage the advantages of regression models like gradient boosting trees. We use *NLPerf* as a competitive baseline in this work. Note that *NLPerf* was shown to outperform several simple mean-value baselines, namely the average over the performances of all available test instances; the average over shared source languages; and the average over models if multiple models are used. This approach can easily incorporate additional features of the experiments, but they do not provide a direct measure of uncertainty.

In Ye et al. (2021) methods used for performance prediction are based on tensor-regression-based ap-

proaches being robust PCA and CP decomposition. Those are developed to provide a more fine-grained performance measure. While confidence intervals in the frequentist sense are provided, no Bayesian analysis was conducted. In Srinivasan et al. (2021) collective MF was used for performance prediction of massive multilingual language models, where it underperformed XGBoost. However, no experiments were shown for bilingual tasks. We differ in our work because we explore the framework of MF extensively and are able to provide a range of experiments contrasting probabilistic and non-probabilistic approaches. Additionally our approach can give CIs as a measure of uncertainty of our predictions. Closer to our approach is the work of Elsahar and Gallé (2019), where instead of evaluating domain similarity or diversity, a linear classifier is used to predict the performance drop under domain shift using metrics based on $\mathcal{H}-$divergence, reverse classification accuracy and confidence measures. In contrast to this body of literature, our methods provide the additional benefit of Bayesian uncertainty quantification.

An area of application for performance prediction, to be additionally highlighted here, is data selection, during which the method can be used to identify a suitable training dataset under domain shift, which will lead to the best model performance. Earlier work in this field in e.g. Blitzer et al. (2007) uses the unsupervised $\mathcal{A}-$distance measure of divergence between domains, while existing literature on phrase based machine translation in Axelrod et al. (2011) and Moore and Lewis (2010) show that perplexity- or cross-entropy based scoring methods are beneficial to select suitable sentences for training, increasing the overall model performance. Connected to this stream of literature is also the work of Ruder and Plank (2017). They present a Bayesian optimisation approach, which is model-independent and is used to learn data selection measures for transfer learning. Additionally, incorporating not only information about the data set, but also about the model, Atwell et al. (2022) shows that the $h-$discrepancy, which is a generalisation of the source guided discrepancy (Kuroki et al., 2019), can be used to identify the best generalization performance of discourse models.

## 5 Conclusion

In this work, we presented an extensive study of various MF methods applied to the problem of performance prediction. Using a Bayesian approach we can give a measure of uncertainty in terms of CIs in addition to point estimates. Additionally, we show that leveraging the obtained bounds of the CIs can guide decision-making regarding whether it is lucrative to deploy a model for certain language-pairs and whether the corresponding datasets for the language-pairs should be acquired.

Our results confirm that bilinear MF methods can be used to predict the performance scores of various NLP tasks, which is computationally less expensive than using a non-linear model like XGboost in *NLPerf*. Furthermore, we show that the MF framework is suited to predict performance scores in cold-start scenarios.

The MF framework was not able to outperform the SOTA for the two-model scenario BLI, which might be due to the small matrix size and unsuitable features. However, BPMF CTX clearly outperformed LR. This suggests that further studies are required regarding non-linear MF methods.

While we chose to present a specific set of tasks within this work, the proposed MF methods can be used for other NLP scenarios where the considered problem can be modelled as a matrix.

## Limitations

While a variety of NLP problems can be modelled as a matrix as our investigations in this paper show, this does not apply to all existing NLP tasks and excludes our methods in its current form from being used on datasets, provided for tasks like e.g. universal dependency parsing or morphological analysis as done in Xia et al. (2020). The reason is, that a suitable matrix representation is not possible.

While we were able to show that context features significantly improve the estimation results, we found that language features did not improve the predictions. Therefore, further studies are necessary to understand how language information can be additionally incorporated into the MF framework for performance prediction.

Our Bayesian approach uses MCMC sampling to perform approximate inference, a process that can be rather time consuming and could be replaced by more efficient sampling methods in the future. Furthermore, our methods are currently limited to linear approaches, for a fair comparison with the non-linear model *NLPerf*, non-linear methods needs to be investigated.

## Acknowledgements

We thank Tianzhi Li for the code basis provided for the non-probabilistic MF approach. This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

## References

Antonios Anastasopoulos and Graham Neubig. 2019. Should all cross-lingual embeddings speak english? *arXiv preprint arXiv:1911.03058*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. *arXiv preprint arXiv:1805.06297*.

Katherine Atwell, Antony Sicilia, Seong Jae Hwang, and Malihe Alikhani. 2022. The change that matters in discourse parsing: Estimating the impact of domain shift on parser error. *Findings of ACL*, pages 824–845.

Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(11):397–422.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of EMNLP*, pages 355–362.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

John Barnard, Robert McCulloch, and Xiao-Li Meng. 2000. Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. *Statistica Sinica*, pages 1281–1311.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, pages 440–447.

Ricardo Cabral, Fernando De la Torre, João P Costeira, and Alexandre Bernardino. 2013. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *Proceedings of the IEEE ICCV*, pages 2488–2495.

George Casella and Roger L. Berger. 2021. *Statistical inference*. Cengage Learning.

Baiyu Chen, Zi Yang, and Zhouwang Yang. 2018a. An algorithm for low-rank matrix factorization and its applications. *Neurocomputing*, 275:1012–1020.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the SIGKDD*, pages 785–794.

Wen-Hao Chen, Chin-Chi Hsu, Yi-An Lai, Vincent Liu, Mi-Yen Yeh, and Shou-De Lin. 2018b. Attribute-aware collaborative filtering: survey and classification. *arXiv preprint arXiv:1810.08765*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Ayan Das and Sudeshna Sarkar. 2020. A survey of the model transfer approaches to cross-lingual dependency parsing. *ACM TALLIP*, 19(5):1–60.

Wietse de Vries, Martijn Wieling, and Malvina Nissim. 2022. Make the best of cross-lingual transfer: Evidence from POS tagging with over 100 languages. In *Proceedings of the ACL (Volume 1: Long Papers)*, pages 7676–7685.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Ke-Lin Du and Madisetti NS Swamy. 2013. *Neural networks and statistical learning*. Springer Science & Business Media.

Hady Elsahar and Matthias Gallé. 2019. To annotate or not? predicting performance drop under domain shift. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2163–2173.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Barry Haddow, Rachel Bawden, Antonio Valerio Miceli Barone, Jindřich Helcl, and Alexandra Birch. 2022. Survey of low-resource machine translation. *Computational Linguistics*, pages 1–67.

Luiz Hespanhol, Caio Sain Vallio, Lucíola Menezes Costa, and Bruno T Saragiotto. 2019. Understanding and interpreting confidence and credible intervals around effect estimates. *Brazilian Journal of Physical Therapy*, 23(4):290–301.

Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. 2012. Prediction of learning curves in machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22–30.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Seiichi Kuroki, Nontawat Charoenphakdee, Han Bao, Junya Honda, Issei Sato, and Masashi Sugiyama. 2019. Unsupervised domain adaptation based on source-guided discrepancy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4122–4129.

Daniel Lewandowski, Dorota Kurowicka, and Harry Joe. 2009. Generating random correlation matrices based on vines and extended onion method. *Journal of multivariate analysis*, 100(9):1989–2001.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, et al. 2019. Choosing transfer languages for cross-lingual learning. *arXiv preprint arXiv:1905.12688*.

Patrick Littell, David R Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL) (Volume 2: Short Papers)*, pages 8–14.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*.

Andriy Mnih and Russ R Salakhutdinov. 2007. Probabilistic matrix factorization. *Advances in neural information processing systems*, 20.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 220–224.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. 2018a. When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323*.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. 2018b. When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323*.

Kenneth Rice and Lingbo Ye. 2022. Expressing regret: a unified view of credible intervals. *The American Statistician*, pages 1–9.

Shruti Rijhwani, Jiateng Xie, Graham Neubig, and Jaime Carbonell. 2019. Zero-shot neural transfer for cross-lingual entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6924–6931.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with Bayesian optimization. *arXiv preprint arXiv:1707.05246*.

Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 880–887.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1351–1361.

Adrian Seyboldt. 2019. Uses of LKJCholeskyCov and LKJCorr. https://discourse.pymc.io/t/uses-of-lkjcholeskycov-and-lkjcorr/3629/2. [Online; accessed 19-October-2022].

Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The cost of training nlp models: A concise overview. *arXiv preprint arXiv:2004.08900*.

Anirudh Srinivasan, Gauri Kholkar, Rahul Kejriwal, Tanuja Ganu, Sandipan Dandapat, Sunayana Sitaram, Balakrishnan Santhanam, Somak Aditya, Kalika Bali, and Monojit Choudhury. 2022. Litmus predictor: An ai assistant for building reliable, high-performing and fair multilingual nlp systems. In *Thirty-sixth AAAI Conference on Artificial Intelligence*.

Anirudh Srinivasan, Sunayana Sitaram, Tanuja Ganu, Sandipan Dandapat, Kalika Bali, and Monojit Choudhury. 2021. Predicting the performance of multilingual nlp models. *arXiv preprint arXiv:2110.08875*.

The PyMC Development Team. 2018. LKJ Cholesky Covariance Priors for Multivariate Normal Models. https://docs.pymc.io/en/v3/pymc-examples/examples/case_studies/LKJ.html. [Online; accessed 19-October-2022].

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8625–8646.

Zihuiwen Ye, Pengfei Liu, Jinlan Fu, and Graham Neubig. 2021. Towards more fine-grained and reliable nlp performance prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Henry Shaowu Yuchi, Simon Mak, and Yao Xie. 2022. Bayesian uncertainty quantification for low-rank matrix completion. *Bayesian Analysis*, 1(1):1–28.

| Task | $\text{MT}^{\text{Wiki}}$ | $\text{Parsing}^{\text{TSF}}$ | $\text{EL}^{\text{TSF}}$ | $\text{POS}^{\text{TSF}}$ | $\text{MT}^{\text{TSF}}$ | BLI |
|---|---|---|---|---|---|---|
| MF $D = 10$ | 5 min | 4 min | 2 min | 7 min | 14 min | 1 min |
| MF $D = 20$ | 5 min | 4 min | 2 min | 7 min | 14 min | 1 min |
| MF CTX $D = 10$ | 8 min | 6 min | 4 min | 12 min | 22 min | 2 min |
| MF CTX $D = 20$ | 8 min | 7 min | 4 min | 12 min | 22 min | 2 min |
| PMF $D = 10$ | 1 h 31 min | 1 h 6 min | 48 min | 1 h 34 min | 1 h 29 min | 1 h 58 min |
| PMF $D = 20$ | 1 h 29 min | 1 h 17 min | 54 min | 2 h 10 min | 2 h 33 min | 1 h 53 min |
| BPMF $D = 10$ | 7 h 40 min | 9 h 54 min | 8 h 52 min | 10 h 8 min | 10 h 10 min | 22 h 2 min |
| BPMF $D = 20$ | 8 h 11 min | 10 h 44 min | 10 h 17 min | 11 h 30 min | 9 h 30 min | 17 h 4 min |
| BPMF $D = 30$ | 6 h 54 min | 10 h 49 min | 10 h 25 min | 11 h 1 min | 9 h 57 min | 18 h 50 min |
| BPMF CTX $D = 10$ | 9 h 48 min | 10 h 33 min | 10 h 22 min | 11 h 34 min | 13 h 27 min | 14 h 25 min |
| BPMF CTX $D = 20$ | 9 h 56 min | 11 h 57 min | 10 h 52 min | 15 h 22 min | 13 h 2 min | 15 h 32 min |
| BPMF CTX $D = 30$ | 9 h 14 min | 12 h 42 min | 11 h 31 min | 14 h 23 min | 13 h 25 min | 15 h 38 min |

Table 5: Execution time required for one training and inference run on average per run for each experiment on one cpu, without hyperparamter tuning, assuming optimal hyperparameters have been found before. Training MF and MF CTX: 5 fold CV.

## A Models Considered in the Experiments

The MT model behind the $\text{MT}^{\text{Wiki}}$ data is trained on aligned Wikipedia data (Schwenk et al., 2021). The training data for translation are WikiMatrix bitexts (Schwenk et al., 2021) containing a mixture of high and low resource languages, mined using an approach based on multilingual sentence embeddings, and tested using NMT (fairseq[4], Transformer) on the TED test set (Qi et al., 2018a) using BLEU performance. The BLI performance dataset is obtained based on the following procedure. Multilingual semi-supervised word embeddings (Conneau et al., 2017), and fully unsupervised cross-lingual word embeddings (Artetxe et al., 2018) for high and low resource languages, which are learnt under a bilingual setting, are tested on BLI using accuracy as the performance measure (Anastasopoulos and Neubig, 2019). For all TSF tasks, the performance scores are obtained using LangRank (Lin et al., 2019), which gives a rank to each transfer language. Therefore, additionally a feature showing the *rank* of each transfer language can be leveraged. In the following a short description about the underlying models is given, a more comprehensive explanation can be found in (Lin et al., 2019). For $\text{MT}^{\text{TSF}}$ the underlying trained model is an attention-based sequence-to-sequence model (Bahdanau et al., 2015). Training is performed on the multilingual TED talk corpus (Qi et al., 2018b). For the $\text{EL}^{\text{TSF}}$ performance scores two character-level LSTM encoders are trained (Rijhwani et al., 2019). The $\text{POS}^{\text{TSF}}$ dataset is obtained through the training of a bi-directional LSTM-CNNs-CRF model (Ma and Hovy, 2016), while the dependency parsers uses a deep biaffine attentional graph-based model (Dozat and Manning, 2016). The diversity of the underlying models, namely being statistical-, neural network- or dictionary based, gives us a better impression about the usability of our predictor.

[4] https://fairseq.readthedocs.io/en/latest/

## B On Computational Complexity

Non-Bayesian MF models have linear time complexity in the number of latent factors $O(D)$ per SGD iteration or $O(D|\delta(\mathbf{R})|)$ for one pass of all observed source-language pairs. The Bayesian MF models have cubic complexity on the size of the latent factors and linear on the number of source/target languages. Due to very small sizes of our datasets, computations were possible on a single cpu. The execution times per model per one run of training and inference are given in Table 5. Compared to the Bayesian approaches, the Non-Bayesian MF methods have a short execution time. It increases with the size of the score matrix, e.g. the model for $\text{MT}^{\text{TSF}}$ with a score matrix size of $54 \times 54$ has the highest number of cells compared to all other models and takes also the longest time to perform one training and inference run. The times shown can be taken as a reference point. Furthermore, implementing the experiments on GPU would speed up the process further, in our case we report the execution times on CPU for a better comparison. The CPU used for our experiments is Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz. Implementing MCMC sampling using GPU will lead to a further execution time improvement.