Parameter Space Factorization for Zero-Shot Learning across Tasks and Languages

Edoardo M. Ponti\textsuperscript{c}, Ivan Vuli\textsuperscript{c}, Ryan Cotterell\textsuperscript{c,ζ}, Marinela Parovi\textsuperscript{c}, Roi Reichart\textsuperscript{τ}, Anna Korhonen\textsuperscript{ι}

\textsuperscript{c}University of Cambridge \textsuperscript{ζ}ETH Zürich \textsuperscript{τ}Technion, IIT
\{ep490,iv250,rdc42,mp939,alk23\}@cam.ac.uk
\textsuperscript{τ}roiri@ie.technion.ac.il

Abstract

Most combinations of NLP tasks and language varieties lack in-domain examples for supervised training because of the paucity of annotated data. How can neural models make sample-efficient generalizations from task–language combinations with available data to low-resource ones? In this work, we propose a Bayesian generative model for the space of neural parameters. We assume that this space can be factorized into latent variables for each language and each task. We infer the posteriors over such latent variables based on data from seen task–language combinations through variational inference. This enables zero-shot classification on unseen combinations at prediction time. For instance, given training data for named entity recognition (NER) in Vietnamese and for part-of-speech (POS) tagging in Wolof, our model can perform accurate predictions for NER in Wolof.

In particular, we experiment with a typologically diverse sample of 33 languages from 4 continents and 11 families, and show that our model yields comparable or better results than state-of-the-art, zero-shot cross-lingual transfer methods. Our code is available at github.com/cambridgeltl/parameter-factorization.

1 Introduction

The annotation efforts in NLP have achieved impressive feats, such as the Universal Dependencies (UD) project (Nivre et al., 2019), which now includes 83 languages. But even UD covers only a meager subset of the world’s estimated 8,506 languages (Hammarström et al., 2016). What is more, the Association for Computational Linguistics Wiki\textsuperscript{1} lists 24 separate NLP tasks. Labeled data, which is both costly and labor-intensive, is missing for many of such task–language combinations. This shortage hinders the development of computational models for the majority of the world’s languages (Snyder and Barzilay, 2010; Ponti et al., 2019a).

A common solution is transferring knowledge across domains, such as tasks and languages (Yogatama et al., 2019; Talmor and Berant, 2019), which holds promise to mitigate the lack of training data inherent to a large spectrum of NLP applications (Täckström et al., 2012; Agić et al., 2016; Ammar et al., 2016; Ponti et al., 2018; Ziser and Reichart, 2018, \textit{inter alia}). In the most extreme scenario, zero-shot learning, no annotated examples are available for the target domain. In particular, zero-shot transfer across languages implies a change in the data domain, and leverages information from resource-rich languages to tackle the same task in a previously unseen target language (Lin et al., 2019; Rijhwani et al., 2019; Artetxe and Schwenk, 2019; Ponti et al., 2019a, \textit{inter alia}). Zero-shot transfer across tasks within the same language (Ruder et al., 2019a), on the other hand, implies a change in the space of labels.

As our main contribution, we propose a Bayesian generative model of the neural parameter space. We assume this to be structured, and for this reason factorizable into task- and language-specific latent variables.\textsuperscript{2} By performing transfer of knowledge from both related tasks \textit{and} related languages (i.e., from \textit{seen} combinations), our model allows for zero-shot prediction on \textit{unseen} task–language combinations. For instance, the

\textsuperscript{1}aclweb.org/aclwiki/State_of_the_art.

\textsuperscript{2}By latent variable we mean every variable that has to be inferred from observed (directly measurable) variables. To avoid confusion, we use the terms \textit{seen} and \textit{unseen} when referring to different task–language combinations.
availability of annotated data for part-of-speech (POS) tagging in Wolof and for named-entity recognition (NER) in Vietnamese supplies plenty of information to infer a task-agnostic representation for Wolof and a language-agnostic representation for NER. Conditioning on these, the appropriate neural parameters for Wolof NER can be generated at evaluation time. While this idea superficially resembles matrix completion for collaborative filtering (Mnih and Salakhutdinov, 2008; Dziugaite and Roy, 2015), the neural parameters are latent and are non-identifiable. Rather than recovering missing entries from partial observations, in our approach we reserve latent variables to each language and each task to tie together neural parameters for combinations that have either of them in common.

We adopt a Bayesian perspective towards inference. The posterior distribution over the model’s latent variables is approximated through stochastic variational inference (Hoffman et al., 2013, SVI). Given the enormous number of parameters, we also explore a memory-efficient inference scheme based on a diagonal plus low-rank approximation of the covariance matrix. This guarantees that our model remains both expressive and tractable.

We evaluate the model on two sequence labeling tasks: POS tagging and NER, relying on a typologically representative sample of 33 languages from 4 continents and 11 families. The results clearly indicate that our generative model surpasses standard baselines based on cross-lingual transfer 1) from the (typologically) nearest source language; 2) from the source language with the most abundant in-domain data (English); and 3) from multiple source languages, in the form of either a multi-task, multi-lingual model with parameter sharing (Wu and Dredze, 2019) or an ensemble of task- and language-specific models (Rahimi et al., 2019).

2 Bayesian Generative Model

In this work, we propose a Bayesian generative model for multi-task, multi-lingual NLP. We train a single Bayesian neural network for several tasks and languages jointly. Formally, we consider a set $T = \{t_1, \ldots, t_n\}$ of $n$ tasks and a set $L = \{l_1, \ldots, l_m\}$ of $m$ languages. The core modeling assumption we make is that the parameter space of the neural network is structured: Specifically, we posit that certain parameters correspond to tasks and others correspond to languages. This structure assumption allows us to generalize to unseen task–language pairs. In this regard, the model is reminiscent of matrix factorization as applied to collaborative filtering (Mnih and Salakhutdinov, 2008; Dziugaite and Roy, 2015).

We now describe our generative model in three steps that match the nesting level of the plates in the diagram in Figure 1. Equivalently, the reader can follow the nesting level of the for loops in Algorithm 1 for an algorithmic illustration of the generative story.

1) **Sampling Task and Language Representations**: To kick off our generative process, we first sample a latent representation for each of the tasks and languages from multivariate Gaussians: $t_i \sim \mathcal{N}(\mu_{t_i}, \Sigma_{t_i}) \in \mathbb{R}^h$ and $l_j \sim \mathcal{N}(\mu_{l_j}, \Sigma_{l_j}) \in \mathbb{R}^h$, respectively. While we present the model in its most general form, we take $\mu_{t_i} = \mu_{l_j} = 0$ and $\Sigma_{t_i} = \Sigma_{l_j} = I$ for the experimental portion of this paper.

2) **Sampling Task–Language-specific Parameters**: Afterward, to generate task–language-specific neural parameters, we sample $\theta_{ij}$ from $\mathcal{N}(f_\psi(t_i, l_j), \text{diag}(f_\phi(t_i, l_j))) \in \mathbb{R}^d$ where $f_\psi(t_i, l_j)$ and $f_\phi(t_i, l_j)$ are learned deep feed-forward neural networks $f_\psi : \mathbb{R}^h \rightarrow \mathbb{R}^d$ and $f_\phi : \mathbb{R}^h \rightarrow \mathbb{R}^d_{\geq 0}$ parametrized by $\psi$ and $\phi$, respectively, similar to Kingma and Welling (2014). These transform the
latent representations into the mean $\mu_{\theta_{ij}}$ and diagonal of the covariance matrix $\sigma_{\theta_{ij}}^2$ for the parameters $\theta_{ij}$ associated with $t_i$ and $l_j$. The feed-forward network $f_\psi$ just has a final linear layer as the mean can range over $\mathbb{R}^d$ whereas $f_\phi$ has a final softplus (defined in Section 3) layer to ensure it ranges only over $\mathbb{R}_{\geq 0}^d$. Following Stolee and Patterson (2019), the networks $f_\psi$ and $f_\phi$ take as input a linear function of the task and language vectors: $t \oplus l \oplus (t \odot l)$, where $\oplus$ stands for concatenation and $\odot$ for element-wise multiplication. The sampled neural parameters $\theta_{ij}$ are partitioned into a weight $W_{ij} \in \mathbb{R}^{c \times c}$ and a bias $b_{ij} \in \mathbb{R}^c$, and reshaped appropriately. Hence, the dimensionality of the Gaussian is chosen to reflect the number of parameters in the affine layer, $d = e \cdot c + c$, where $e$ is the dimensionality of the input token embeddings (detailed in the next paragraph) and $c$ is the maximum number of classes across tasks. The number of hidden layers and the hidden size of $f_\psi$ and $f_\phi$ are hyper-parameters discussed in Section 4.2. We tie the parameters $\psi$ and $\phi$ for all layers except for the last to reduce the parameter count. We note that the space of parameters for all tasks and languages forms a tensor $\Theta \in \mathbb{R}^{n \times m \times d}$, where $d$ is the number of parameters of the largest model.

### Sampling Task Labels
Finally, we sample the $k^{\text{th}}$ label $y_{ijk}$ for the $j^{\text{th}}$ task and the $j^{\text{th}}$ language from a final softmax: $p(y_{ijk} \mid x_{ijk}, \theta_{ij}) = \text{softmax}(W_{ij} \text{BERT}(x_{ijk}) + b_{ij})$ where $\text{BERT}(x_{ijk}) \in \mathbb{R}^c$ is the multi-lingual BERT (Pires et al., 2019) encoder. The incorporation of m-BERT as a pre-trained multi-lingual embedding allows for enhanced cross-lingual transfer.

Consider the Cartesian product of all tasks and languages $T \times L$. We can decompose this product into seen task–language pairs $S$ and unseen task–language pairs $U$, i.e., $T \times L = S \sqcup U$. Naturally, we are only able to train our model on the seen task–language pairs $S$. However, as we estimate all task–language parameter vectors $\theta_{ij}$ jointly, our model allows us to draw inferences about the parameters for pairs in $U$ as well. The intuition for why this should work is as follows: By observing multiple pairs where the task (language) is the same but the language (task) varies, the model learns to distill the relevant knowledge for zero-shot learning because our generative model structurally enforces a disentangled representations —separating representations for the tasks from the representations for the languages rather than lumping them together into a single entangled representation (Wu and Dredze, 2019, inter alia). Furthermore, the neural networks $f_\psi$ and $f_\phi$ mapping the task- and language-specific latent variables to neural parameters are shared, allowing the model to generalize across task–language pairs.

### 3 Variational Inference

Exact computation of the posterior over the latent variables $p(\theta, t, l \mid x)$ is intractable. Thus, we need to resort to an approximation. In this work, we consider variational inference as our approximate inference scheme. Variational inference finds an approximate posterior over the latent variables by minimizing the variational gap, which may be expressed as the Kullback–Leibler (KL) divergence between the variational approximation $q(\theta, t, l)$ and the true posterior $p(\theta, t, l \mid x)$. In our work, we employ the following variational distributions:

$$q_A = \mathcal{N}(m_t, S_t) \quad m_t \in \mathbb{R}^h, S_t \in \mathbb{R}^{h \times h} \quad (1)$$
$$q_V = \mathcal{N}(m_l, S_l) \quad m_l \in \mathbb{R}^h, S_l \in \mathbb{R}^{h \times h} \quad (2)$$
$$q_E = \mathcal{N}(f_\phi(t, l), \text{diag}(f_\phi(t, l))) \quad (3)$$

We note the unusual choice to tie parameters between the generative model and the variational

---

**Algorithm 1** Generative Model of Neural Parameters for Multi-task, Multi-lingual NLP.

```plaintext
1: for $t_i \in T$:
   2: $t_i \sim \mathcal{N}(\mu_{t_i}, \Sigma_{t_i})$
3: for $l_j \in L$:
   4: $l_j \sim \mathcal{N}(\mu_{l_j}, \Sigma_{l_j})$
5: for $t_i \in T$:
   6: for $l_j \in L$:
   7: $\mu_{\theta_{ij}} = f_\psi(t_i, l_j)$
   8: $\Sigma_{\theta_{ij}} = f_\phi(t_i, l_j)$
   9: $\theta_{ij} \sim \mathcal{N}(\mu_{\theta_{ij}}, \Sigma_{\theta_{ij}})$
10: for $x_{ijk} \in X_{ij}$:
11: $y_{ijk} \sim p(\cdot \mid x_{ijk}, \theta_{ij})$
```

---

3 Different tasks might involve different class numbers; the number of parameters hence oscillates. The extra dimensions not needed for a task can be considered as padded with zeros.
\[
\begin{align*}
\text{KL}(q(\theta, t, l) \| p(\theta, t, l | x)) &= - \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} \log \frac{p(\theta, t, l | x)}{q(\theta, t, l)} \\
&= - \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} [\log p(\theta, t, l, x) - \log p(x) - \log q(\theta, t, l)] \\
&= \log p(x) - \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} \log \frac{p(\theta, t, l, x)}{q(\theta, t, l)} \triangleq \log p(x) - \mathcal{L} \quad (4)
\end{align*}
\]

\[
\log p(x) = \log \left( \int \int \int p(x, \theta, t, l) \, d\theta \, dt \, dl \right)
= \log \left( \int \int \int q_\lambda(t) q_\nu(l) q_\xi(\theta | t, l) p(x | \theta) p(\theta | t, l) p(t) p(l) \, d\theta \, dt \, dl \right)
= \log \left( \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} \frac{p(\theta, t, l, x)}{q_\lambda(t) q_\nu(l) q_\xi(\theta | t, l)} \right) \\
\geq \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} \left[ \log p(x | \theta) + \log \frac{p(\theta, t, l, x)}{q_\lambda(t) q_\nu(l) q_\xi(\theta | t, l)} + \log p(t) + \log p(l) \right] \\
= \mathbb{E} \mathbb{E} \mathbb{E}_{t \sim q_t, t \sim q_{\theta}, \theta \sim q_\theta} \left( \text{KL}(q_\lambda(t) \| p(t)) + \text{KL}(q_\nu(l) \| p(l)) + \text{KL}(q_\xi(\theta | t, l) \| p(\theta | t, l)) \right)
\]

\[
\text{requires approximation}
\]
\[
\text{closed-form solution}
\]

(5)

(6)

family in Equation (3); however, we found that this choice performs better in our experiments.

Through a standard algebraic manipulation in Equation (4), the KL-divergence for our generative model can be shown to equal the marginal log-likelihood \( \log p(x) \), independent from \( q(\cdot) \), and the so-called evidence lower bound (ELBO) \( \mathcal{L} \). Thus, approximate inference becomes an optimization problem where maximizing \( \mathcal{L} \) results in minimizing the KL-divergence. One derives \( \mathcal{L} \) by expanding the marginal log-likelihood as in Equation (5) by means of Jensen’s inequality. We also show that \( \mathcal{L} \) can be further broken into a series of terms as illustrated in Equation (6). In particular, we see that it is only the first term in the expansion that requires approximation. The subsequent terms are KL-divergences between variational and true distributions that have closed-form solution due to our choice of prior. Due to the parameter-tying scheme above, the KL-divergence in Equation (6) between the variational distribution \( q_\xi(\theta | t, l) \) and the prior distribution \( p(\theta | t, l) \) is zero.

In general, the covariance matrices \( S_t \) and \( S_l \) in Equation (1) and Equation (2) will require \( O(h^2) \) space to store. As \( h \) is often very large, it is impractical to materialize either matrix in its entirety. Thus, in this work, we experiment with smaller matrices that have a reduced memory footprint; specifically, we consider a diagonal covariance matrix and a diagonal plus low-rank covariance structure. A diagonal covariance matrix makes computation feasible with a complexity of \( O(\cdot) \); this, however, comes at the cost of not letting parameters influence each other, and thus failing to capture their complex interactions. To allow for a more expressive variational family, we also consider a covariance matrix that is the sum of a diagonal matrix and a low-rank matrix:

\[
S(t) = \text{diag}(\delta_t^2) + B_t B_t^\top \quad (7)
\]

\[
S_l = \text{diag}(\delta_l^2) + B_l B_l^\top \quad (8)
\]
where $B \in \mathbb{R}^{k \times k}$ ensures that $\text{rank}(BB^T) \leq k$ and $\text{diag}(\delta)$ is diagonal. We can store this structured covariance matrix in $O(kh)$ space.

By definition, covariance matrices must be symmetric and positive semi-definite. The first property holds by construction. The second property is enforced by a softplus parameterization where $\text{softplus}(\cdot) \triangleq \ln(1 + \exp(\cdot))$. Specifically, we define $\delta^2 = \text{softplus}(\rho)$ and we optimize over $\rho$.

### 3.1 Stochastic Variational Inference

To speed up the training time, we make use of stochastic variational inference (Hoffman et al., 2013). In this setting, we randomly sample a task $t_i \in T$ and language $l_j \in L$ among seen combinations during each training step, and randomly select a batch of examples from the dataset for the sampled task–language pair. We then optimize the parameters of the feed-forward neural networks $\psi$ and $\phi$ as well as the parameters of the variational approximation to the posterior $m_i, m_i, \rho_i, \rho_i, B_i$, and $B_i$ with a stochastic gradient-based optimizer (discussed in Section 4.2).

The KL divergence terms and their gradients in the ELBO appearing in Equation (6) can be computed in closed form as the relevant densities are Gaussian (Duchi, 2007, p. 13). Moreover, they can be calculated for Gaussians with diagonal and diagonal plus low-rank covariance structures without explicitly unfolding the full matrix. For a choice of prior $p = N(0, I)$ and a diagonal plus low-rank covariance structure, we have:

$$
\text{KL}(q \| p) = \frac{1}{2} \left[ \sum_{i=1}^{h} (m_i^2 + \delta_i^2) + \sum_{j=1}^{k} b_{ij}^2 \right] - h - \ln \text{det}(S)
$$

where $b_{ij}$ is the element in the $i$-th row and $j$-th column of $B$. The last term can be estimated without computing the full matrix explicitly thanks to the generalization of the matrix–determinant lemma, which, applied to the factored covariance structure, yields:

$$
\ln \text{det}(S) = \ln \left[ \text{det}(I + B^T \text{diag}(\delta^{-2})B) \right] + \sum_{i=1}^{h} \ln(\delta_i^2)
$$

(10)

where $I \in \mathbb{R}^k$. The KL divergence for the variant with diagonal covariance is just a special case of Equation (9) with $b_{ij} = 0$.

However, as stated before, the following expectation does not admit a closed-form solution. Thus we consider a Monte Carlo approximation:

$$
\mathbb{E}_{\theta \sim q_\xi} \log p(x \mid \theta) = \int q_\xi(\theta) \log p(x \mid \theta) \, d\theta \\
\approx \frac{1}{V} \sum_{v=1}^{V} \log p(x \mid \theta^{(v)}) \quad \text{where } \theta^{(v)} \sim q_\xi
$$

(11)

where $V$ is the number of Monte Carlo samples taken. In order to allow the gradient to easily flow through the generated samples, we adopt the re-parametrization trick (Kingma and Welling, 2014). Specifically, we exploit the following identities $t_i = \mu_i + \sigma_i \odot \epsilon$ and $l_j = \mu_j + \sigma_j \odot \epsilon$, where $\epsilon \sim N(0, I)$ and $\odot$ is the Hadamard product. For the diagonal plus low-rank covariance structure, we exploit the identity:

$$
\mu + \text{diag}(\delta^2 \odot \epsilon) + B \zeta
$$

(12)

where $\epsilon \in \mathbb{R}^h$, $\zeta \in \mathbb{R}^k$, and both are sampled from $N(0, I)$. The mean $\mu_{\theta_{ij}}$ and the diagonal of the covariance matrix $\sigma^2_{\theta_{ij}}$ are deterministically computed given the above samples and the parameters $\theta_{ij}$ are sampled from $N(\mu_{\theta_{ij}}, \text{diag}(\sigma^2_{\theta_{ij}}))$.

### 3.2 Posterior Predictive Distribution

During test time, we perform zero-shot predictions on an unseen task–language pair by plugging in the posterior means (under the variational approximation) into the model. As an alternative, we experimented with ensemble predictions through Bayesian model averaging. That is, for data for seen combinations $x_S$ and data for unseen combinations $x_U$, the true predictive posterior can be approximated as $p(x_U \mid x_S) = \int p(x_U \mid \theta, x_S) q_\xi(\theta \mid x_S) \, d\theta \approx \sum_{v=1}^{V} p(x_U \mid \theta^{(v)}, x_S)$, where $V$ are 100 Monte Carlo samples from the posterior $q_\xi$. Performances on the development sets are comparable to simply plugging in the posterior mean.
4 Experimental Setup

4.1 Data

We select NER and POS tagging as our experimental tasks because their datasets encompass an ample and diverse sample of languages, and are common benchmarks for resource-poor NLP (Cotterell and Duh, 2017, inter alia). In particular, we opt for WikiANN (Pan et al., 2017) for the NER task and Universal Dependencies 2.4 (UD; Nivre et al., 2019) for POS tagging. Our sample of languages is chosen from the intersection of those available in WikiANN and UD. However, we remark that this sample is heavily biased towards the Indo-European family (Gerz et al., 2018). Instead, the selection should be: i) typologically diverse, to ensure that the evaluation scores truly reflect the expected cross-lingual performance (Ponti et al., 2020); ii) a mixture of resource-rich and low-resource languages, to recreate a realistic setting and to allow for studying the effect of data size. Hence, we further filter the languages in order to make the sample more balanced. In particular, we sub-sample Indo-European languages by including only resource-poor ones, and keep all the languages from other families. Our final sample comprises 33 languages from 4 continents (17 from Asia, 11 from Europe, 4 from Africa, and 1 from South America) and from 11 families (6 Uralic, 6 Indo-European, 5 Afroasiatic, 3 Niger-Congo, 3 Turkic, 2 Austronesian, 2 Dravidian, 1 Austroasiatic, 1 Kra-Dai, 1 Tupian, 1 Sino-Tibetan), as well as 2 isolates. The full list of language iso 639-2 codes is reported in Figure 2.

In order to simulate a zero-shot setting, we hold out in turn half of all possible task–language pairs and regard them as unseen, while treating the others as seen pairs. The partition is performed in such a way that a held-out pair has data available for the same task in a different language, and for the same language in a different task. Under this constraint, pairs are assigned to train or evaluation at random.\(^6\)

We randomly split the WikiANN datasets into training, development, and test portions with a proportion of 80-10-10. We use the provided splits for UD; if the training set for a language is missing, we treat the test set as such when the language is held out, and as a training set when it is among the seen pairs.\(^8\)

4.2 Hyper-parameters

The multilingual m-BERT encoder is initialized with parameters pre-trained on masked language modeling and next sentence prediction on 104 languages (Devlin et al., 2019). We opt for the cased BERT-BASE architecture, which consists of 12 layers with 12 attention heads and a hidden size of 768. As a consequence, this is also the dimension \(e\) of each encoded WordPiece unit, a subword unit obtained through BPE (Wu et al., 2016). The dimension \(h\) of the multivariate Gaussian for task and language latent variables is set to 100. The deep feed-forward networks \(f_\psi\) and \(f_\phi\) have 6 layers with a hidden size of 400 for the first layer, 768 for the internal layers, and ReLU non-linear activations. Their depth and width were selected based on validation performance.

The expectations over latent variables in Equation (6) are approximated through 3 Monte Carlo samples per batch during training. The KL terms are weighted with \(\frac{1}{K}\) uniformly across training, where \(|K|\) is the number of mini-batches.\(^9\) We initialize all the means of the variational approximation with a random sample from \(\mathcal{N}(0, 0.1)\), and the parameters for covariance matrices \(S\) of the variational approximation with a random sample from \(\mathcal{U}(0, 0.5)\) following Stolee and Patterson (2019). We choose \(k = 10\) as the number of columns of \(B\) so it fits into memory. The maximum sequence length for inputs is limited to 250. The batch size is set to 8, and the best setting for the Adam optimizer (Kingma and Ba, 2015) was found to be an initial learning rate of \(5 \times 10^{-6}\) based on grid search. In order to avoid over-fitting, we perform early stopping with a patience of 10 and a validation frequency of 2.5K steps.

4.3 Baselines

We consider four baselines for cross-lingual transfer that also use BERT as an encoder shared across all languages.\(^8\)

\(^6\)We use the controlled partitioning for the following reason. If a language lacks data both for NER and for POS, the proposed factorization method cannot provide estimates for its posterior. We leave model extensions that can handle such cases for future work.

\(^7\)See Section 5.2 for further experiments on splits controlled for language distance and sample size.

\(^8\)Note that, in the second case, no evaluation takes place on such language.

\(^9\)Available at github.com/google-research/bert/blob/master/multilingual.md.
First Baseline. A common approach is transfer from the nearest source (NS) language, which selects the most compatible source to a target language in terms of similarity. In particular, the selection can be based on family membership (Zeman and Resnik, 2008; Cotterell and Heigold, 2017; Kann et al., 2017), typological features (Deri and Knight, 2016), KL-divergence between part-of-speech trigram distributions (Rosa and Žabokrtský 2015; Agić, 2017), tree edit distance of delexicalized dependency parses (Ponti et al., 2018), or a combination of the above (Lin et al., 2019). In our work, during evaluation, we choose the classifier associated with the observed language with the highest cosine similarity between its typological features and those of the held-out language. These features are sourced from URIEL (Littell et al., 2017) and contain information about family, area, syntax, and phonology.

Second Baseline. We also consider transfer from the largest source (LS) language, that is, the language with most training examples. This approach has been adopted by several recent works on cross-lingual transfer (Conneau et al., 2018; Artetxe et al., 2020, inter alia). In our implementation, we always select the English classifier for prediction. In order to make this baseline comparable to our model, we adjust the number of English NER training examples to the sum of the examples available for all seen languages \( S \).

Third Baseline. Next, we apply a protocol designed by Rahimi et al. (2019) for weighting the predictions of a classifier ensemble according to their reliability. For a specific task, the reliability of each language-specific classifier is estimated through a Bayesian graphical model. Intuitively, this model learns from error patterns, which behave more randomly for untrustworthy models and more consistently for the others. Among the protocols proposed in the paper, we opt for BEA in its zero-shot, token-based version, as it achieves the highest scores in a setting comparable to the current experiment. We refer to the original paper for the details.

Fourth Baseline. Finally, we take inspiration from Wu and Dredze (2019). The joint multilingual (JM) baseline, contrary to the previous baselines, consists of two classifiers (one for POS tagging and another for NER) shared among all observed languages for a specific task. We follow the original implementation of Wu and Dredze (2019), closely adopting all recommended hyper-parameters and strategies, such as freezing the parameters of all baseline networks to match the parameter count is detrimental if the BERT encoder is kept trainable, which was also verified in previous work (Peters et al., 2019).

5 Results and Discussion

5.1 Zero-shot Transfer

Firstly, we present the results for zero-shot prediction based on our generative model using both of the approximate inference schemes (with diagonal covariance PF-d and factor covariance PF-lr). Table 1 summarizes the results on the two tasks of POS tagging and NER averaged across all languages. Our model (in both its variants) outperforms the four baselines on both tasks, including state-of-the-art alternative methods. In particular, PF-d and PF-lr gain 4.49 / 4.20 in accuracy (∼7%) for POS tagging and 7.29 / 7.73 in F1 score (∼10%) for NER on average compared to transfer from the largest source (LS), the strongest baseline for single-source transfer. Compared to multilingual joint transfer from multiple sources (JM), our two variants gain 0.95 / 0.67 in accuracy (∼1%) for POS tagging and +0.61 / +1.05 in F1 score (∼1%).

More details about the individual results on each task-language pair are provided in Figure 2, which includes the mean of the results over 3 separate runs. Overall, we obtain improvements in 23/33 languages for NER and on 27/45 treebanks for POS tagging, which further supports the benefits of transferring both from tasks and languages.

Considering the baselines, the relative performance of LS versus NS is an interesting finding per se. LS largely outperforms NS on both POS...
Table 1: Results per task averaged across all languages.

<table>
<thead>
<tr>
<th>Task</th>
<th>POS</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEA</td>
<td>47.45 ± 0.56</td>
<td>66.45 ± 0.56</td>
</tr>
<tr>
<td>NS</td>
<td>42.14 ± 0.43</td>
<td>64.04 ± 0.18</td>
</tr>
<tr>
<td>LS</td>
<td>60.51 ± 0.43</td>
<td>78.97 ± 0.56</td>
</tr>
<tr>
<td>JM</td>
<td>74.81 ± 0.18</td>
<td>86.05 ± 0.13</td>
</tr>
<tr>
<td>PFD</td>
<td>65.00 ± 0.12</td>
<td>86.70 ± 0.10</td>
</tr>
<tr>
<td>PFD</td>
<td>64.71 ± 0.18</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: Results for NER (top) and POS tagging (bottom): Four baselines for cross-lingual transfer compared to Matrix Factorization with diagonal covariance and diagonal plus low-rank covariance.

Table 1: Results per task averaged across all languages.
tagging and NER. This shows that having more data is more informative than relying primarily on similarity according to linguistic properties. This finding contradicts the received wisdom (Rosa and Žabokrtský, 2015; Cotterell and Heigold, 2017; Lin et al., 2019, *inter alia*) that related languages tend to be the most reliable source. We conjecture that this is due to the pre-trained multi-lingual BERT encoder, which helps to bridge the gap between unrelated languages (Wu and Dredze, 2019).

The two baselines that hinge upon transfer from multiple sources lie on opposite sides of the spectrum in terms of performance. On the one hand, BEA achieves the lowest average score for NER, and surpasses only NS for POS tagging. We speculate that this is due to the following: i) adapting the protocol from Rahimi et al. (2019) to our model implies assigning a separate classifier head to each task–language pair, each of which is exposed to fewer examples compared to a shared one. This fragmentation fails to take advantage of the massively multilingual nature of the encoder; ii) our language sample is more typologically diverse, which means that most source languages are unreliable predictors. On the other hand, JM yields extremely competitive scores. Similarly to our model, it integrates knowledge from multiple languages and tasks. The extra boost in our model stems from its ability to disentangle each aspect of such knowledge and recombine it appropriately.

Moreover, comparing the two approximate inference schemes from Section 3.1, PF-lr obtains a small but statistically significant improvement over PF-d in NER, whereas they achieve the same performance on POS tagging. This means that the posterior is modeled well enough by a Gaussian where covariance among co-variates is negligible.

We see that even for the best model (PF-lr) there is a wide variation in the scores for the same task across languages. POS tagging accuracy ranges from 12.56 ± 4.07 in Guarani to 86.71 ± 0.67 in Galician, and NER F1 scores range from 49.44±0.69 in Amharic to 96.20±0.11 in Upper Sorbian. Part of this variation is explained by the fact that the multilingual BERT encoder is not pre-trained in a subset of these languages (e.g., Amharic, Guarani, Uyghur). Another cause is more straightforward: The scores are expected to be lower in languages for which we have fewer training examples in the seen task–language pairs.

| Task | \(|L| = 11\) | \(|L| = 22\) |
|------|-------------|-------------|
| POS  | 72.44 53.25 | 66.59 63.22 |
| NER  | 89.51 81.73 | 86.78 85.12 |

Table 2: Average performance when relying on \(|L|\) similar (Sim) versus different (Dif) languages in the train and evaluation sets.

5.2 Language Distance and Sample Size

While we designed the language sample to be both realistic and representative of the cross-lingual variation, there are several factors inherent to a sample that can affect the zero-shot transfer performance: i) *language distance*, the similarity between seen and held-out languages; and ii) *sample size*, the number of seen languages. In order to disentangle these factors, we construct subsets of size \(|L|\) so that training and evaluation languages are either maximally similar (Sim) or maximally different (Dif). As a proxy measure, we consider as ‘similar’ languages belonging to the same family. In Table 2, we report the performance of parameter factorization with diagonal plus low-rank covariance (PF-lr), the best model from Section 5.1, for each of these subsets.

Based on Table 2, there emerges a trade-off between language distance and sample size. In particular, performance is higher in Sim subsets compared to Dif subsets for both tasks (POS and NER) and for both sample sizes \(|L| \in \{11, 22\}\). In larger sample sizes, the average performance increases for Dif but decreases for Sim. Intuitively, languages with labeled data for several relatives benefit from small, homogeneous subsets. Introducing further languages introduces noise. Instead, languages where this is not possible (such as isolates) benefit from an increase in sample size.

5.3 Entropy of the Predictive Distribution

A notable problem of point estimate methods is their tendency to assign most of the probability mass to a single class even in scenarios with high uncertainty. Zero-shot transfer is one of such scenarios, because it involves drastic distribution shifts in the data (Rabanser et al., 2019). A key advantage of Bayesian inference, instead, is marginalization over parameters, which yields smoother posterior predictive distributions (Kendall and Gal, 2017; Wilson, 2019).
We run an analysis of predictions based on (approximate) Bayesian model averaging. First, we randomly sample 800 examples from each test set of a task–language pair. For each example, we predict a distribution over classes \( Y \) through model averaging based on 10 samples from the posteriors. We then measure the prediction entropy of each example—that is, \( H(p) = -\sum_y |Y| p(Y = y) \log p(Y = y) \)—whose plot is shown in Figure 3.

Entropy is a measure of uncertainty. Intuitively, the uniform categorical distribution (maximum uncertainty) has the highest entropy, whereas if the whole probability mass falls into a single class (maximum confidence), then the entropy \( H = 0.14 \). As it emerges from Figure 3, predictions in certain languages tend to have higher entropy on average, such as in Amharic, Guaraní, Uyghur, or Assyrian Neo-Aramaic. This aligns well with the performance metrics in Figure 2. In practice, languages with low scores tend to display high entropy in the predictive distribution, as expected. To verify this claim, we measure the Pearson’s correlation between entropies of each task–language pair in Figure 3 and performance metrics. We find a very strong negative correlation with a coefficient of \( \rho = -0.914 \) and a two-tailed p-value of \( 1.018 \times 10^{-26} \).

### 6 Related Work

Our approach builds on ideas from several different fields: cross-lingual transfer in NLP, with a particular focus on sequence labeling tasks, as well as matrix factorization, contextual parameter generation, and neural Bayesian methods.

#### Cross-Lingual Transfer for Sequence Labeling

One of the two dominant approaches for cross-lingual transfer is *projecting annotations* from a source language text to a target language text. This technique was pioneered by Yarowsky et al. (2001) and Hwa et al. (2005) for parsing, and later extended to applications such as POS tagging (Das and Petrov, 2011; Garrette et al., 2013; Täckström et al., 2012; Duong et al., 2014; Huck et al., 2019) and NER (Ni et al., 2017; Enghoff et al., 2018; Agerri et al., 2018; Jain et al., 2019). This requires tokens to be aligned through a parallel corpus, a
machine translation system, or a bilingual dictionary (Durrett et al., 2012; Mayhew et al., 2017). However, creating machine translation and word-alignment systems demands parallel texts in the first place, while automatically induced bilingual lexicons are noisy and offer only limited coverage (Artetxe et al., 2018; Duan et al., 2020). Furthermore, errors inherent to such systems cascade along the projection pipeline (Agić et al., 2015).

The second approach, model transfer, offers higher flexibility (Conneau et al., 2018). The main idea is to train a model directly on the source data, and then deploy it onto target data (Zeman and Resnik 2008). Crucially, bridging between different lexica requires input features to be language-agnostic. While originally this implied delexicalization, replacing words with universal POS tags (McDonald et al., 2011; Dehouck and Denis, 2017), cross-lingual Brown clusters (Täckström et al., 2012; Rasooli and Collins, 2017), or cross-lingual knowledge base grounding through wikification (Camacho-Collados et al., 2016; Tsai et al., 2016), more recently these have been supplanted by cross-lingual word embeddings (Ammar et al. 2016; Zhang et al., 2016; Xie et al., 2018; Ruder et al., 2019b) and multilingual pretrained language models (Devlin et al., 2019; Conneau et al., 2020).

An orthogonal research thread regards the selection of the source language(s). In particular, multi-source transfer was shown to surpass single-best source transfer in NER (Fang and Cohn, 2017; Rahimi et al., 2019) and POS tagging (Enghoff et al., 2018; Plank and Agić, 2018). Our parameter space factorization model can be conceived as an extension of multi-source cross-lingual model transfer to a cross-task setting.

**Data Matrix Factorization.** Although we are the first to propose a factorization of the parameter space for unseen combinations of tasks and languages, the factorization of data for collaborative filtering and social recommendation is an established research area. In particular, the missing values in sparse data structures such as user-movie review matrices can be filled via probabilistic matrix factorization (PMF) through a linear combination of user and movie matrices (Mnih and Salakhutdinov, 2008; Ma et al., 2008; Shan and Banerjee, 2010, *inter alia*) or through neural networks (Dziugaite and Roy, 2015). Inference for PMF can be carried out through MAP inference (Dziugaite and Roy, 2015), Markov chain Monte Carlo (Salakhutdinov and Mnih, 2008) or stochastic variational inference (Stolee and Patterson, 2019). Contrary to prior work, we perform factorization on latent variables (task- and language-specific parameters) rather than observed ones (data).

**Contextual Parameter Generation.** Our model is reminiscent of the idea that parameters can be conditioned on language representations, as proposed by Platanios et al. (2018). However, since this approach is limited to a single task and a joint learning setting, it is not suitable for generalization in a zero-shot transfer setting.

**Bayesian Neural Models.** So far, these models have found only limited application in NLP for resource-poor languages, despite their desirable properties. Firstly, they can incorporate priors over parameters to endow neural networks with the correct inductive biases towards language: Ponti et al. (2019b) constructed a prior imbued with universal linguistic knowledge for zero- and few-shot character-level language modeling. Secondly, they avoid the risk of over-fitting by taking into account uncertainty. For instance, Shareghi et al. (2019) and Doitch et al. (2019) use a perturbation model to sample high-quality and diverse solutions for structured prediction in cross-lingual parsing.

**7 Conclusion**

The main contribution of our work is a Bayesian generative model for multiple NLP tasks and languages. At its core lies the idea that the space of neural weights can be factorized into latent variables for each task and each language. While training data are available only for a meager subset of task–language combinations, our model opens up the possibility to perform prediction in novel, undocumented combinations at evaluation time. We performed inference through stochastic variational methods, and ran experiments on zero-shot named entity recognition (NER) and part-of-speech (POS) tagging in a typologically diverse set of 33 languages. Based on the reported results, we conclude that leveraging the information from tasks and languages simultaneously is superior to model transfer from English (relying on more abundant in-task data in the source language), from the most typologically similar language
(relying on prior information on language relatedness), or from multiple source languages. Moreover, we found that the entropy of predictive posterior distributions obtained through Bayesian model averaging correlates almost perfectly with the error rate in the prediction. As a consequence, our approach holds promise to alleviating data paucity issues for a wide spectrum of languages and tasks, and to make knowledge transfer more robust to uncertainty.

Finally, we remark that our model is amenable to be extended to multilingual tasks beyond sequence labeling—such as natural language inference (Conneau et al., 2018) and question answering (Artetxe et al., 2020; Lewis et al., 2019; Clark et al., 2020)—and to zero-shot transfer across combinations of multiple modalities (e.g., speech, text, and vision) with tasks and languages. We leave these exciting research threads for future research.

Acknowledgments

We would like to thank action editor Jacob Eisenstein and the three anonymous reviewers at TACL. This work is supported by the ERC Consolidator Grant LEXICAL (no 648909) and the Google Faculty Research Award 2018. RR was partially funded by ISF personal grant no. 1625/18.

References

Rodrigo Agerri, Xavier Gómez Guinovart, German Rigau, and Miguel Anxo Solla Portela. 2018. Developing new linguistic resources and tools for the Galician language. In Proceedings of LREC.


Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. Transactions of the ACL, 4:431–444. DOI: https://doi.org/10.1162/tacl_a_00109


John Duchi. 2007. Derivations for linear algebra and optimization, University of California, Berkeley.


Dan Garrette, Jason Mielens, and Jason Baldridge. 2013. Real-world semi-supervised learning of


Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of ACL*, pages 151–164. DOI: https://doi.org/10.18653/v1/P19-1015


Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, and others. 2016, Google’s neural machine translation system: Bridging the gap between human and machine translation, Google.
If both $p \equiv \mathcal{N}(\mu, \Sigma)$ and $q \equiv \mathcal{N}(m, S)$ are multivariate Gaussians, their KL-divergence can be computed analytically as follows:

$$KL(q || p) = \frac{1}{2} \left[ \ln \frac{|S|}{|\Sigma|} - d + \text{tr}(S^{-1} \Sigma) + (m - \mu)^\top S^{-1} (m - \mu) \right]$$  (13)

By substituting $m = 0$ and $S = I$, it is trivial to obtain Equation (9).

### B Visualization of the Learned Posteriors

The approximate posteriors of the latent variables can be visualized in order to study the learned representations for languages. Previous work (Johnson et al., 2017; Östling and Tiedemann, 2017; Malaviya et al., 2017; Bjerva and Augenstein, 2018) induced point estimates of language representations from artificial tokens concatenated to every input sentence, or from the aggregated values of the hidden state of a neural encoder. The information contained in such representations depends on the task (Bjerva and Augenstein, 2018), but mainly reflects the structural properties of each language (Bjerva et al., 2019).

In our work, due to the estimation procedure, languages are represented by full distributions rather than point estimates. By inspecting the learned representations, language similarities do not appear to follow the structural properties of languages. This is most likely due to the fact that parameter factorization takes place after the multilingual BERT encoding, which blends the structural...
differences across languages. A fair comparison with previous works without such an encoder is left for future investigation.

As an example, consider two pairs of languages from two distinct families: Yoruba and Wolof are Niger-Congo from the Atlantic-Congo branch, Tamil and Telugu are Dravidian. We take 1,000 samples from the approximate posterior over the latent variables for each of these languages. In particular, we focus on the variational scheme with a low-rank covariance structure. We then reduce the dimensionality of each sample to 4 through PCA, and we plot the density along each resulting dimension in Figure 4. We observe that density areas of each dimension do not necessarily overlap between members of the same family. Hence, the learned representations depend on more than genealogy.

15Note that the dimensionality reduced samples are also Gaussian since PCA is a linear method.