Evaluating the Values of Sources in Transfer Learning

Md Rizwan Parvez
University of California Los Angeles
rizwan@cs.ucla.edu

Kai-Wei Chang
University of California Los Angeles
kwchang@cs.ucla.edu

Abstract

Transfer learning that adapts a model trained on data-rich sources to low-resource targets has been widely applied in natural language processing (NLP). However, when training a transfer model over multiple sources, not every source is equally useful for the target. To better transfer a model, it is essential to understand the values of the sources. In this paper, we develop SEAL-Shap, an efficient source valuation framework for quantifying the usefulness of the sources (e.g., domains/languages) in transfer learning based on the Shapley value method. Experiments and comprehensive analyses on both cross-domain and cross-lingual transfers demonstrate that our framework is not only effective in choosing useful transfer sources but also the source values match the intuitive source-target similarity.

1 Introduction

Transfer learning has been widely used in learning models for low-resource scenarios by leveraging the supervision provided in data-rich source corpora. It has been applied to NLP tasks in various settings including domain adaptation (Blitzer et al., 2007; Ruder and Plank, 2017), cross-lingual transfer (Täckström et al., 2013; Wu and Dredze, 2019), and task transfer (Liu et al., 2019b; Vu et al., 2020).

A common transfer learning setting is to train a model on a set of sources and then evaluate it on the corresponding target (Yao and Doretto, 2010; Yang et al., 2020).[1] However, not every source corpus contributes equally to the transfer model. Some of them may even cause a performance drop (Ghorbani and Zou, 2019; Lin et al., 2019). Therefore, it is essential to understand the value of each source in the transfer learning not only to achieve

---

[1]In this paper, we focus on two transfer learning scenarios: 1) cross-lingual and 2) cross-domain. We train a model on a set of source corpora and evaluate on a target corpus where each "corpus" refers to the corresponding domain or language.
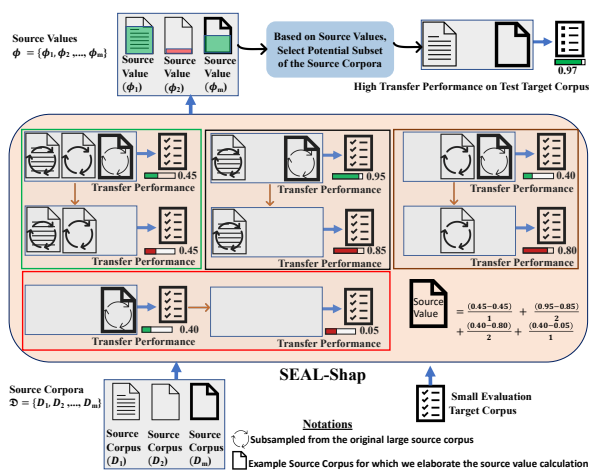

Figure 1: SEAL-Shap estimates the value of each source corpus by the average marginal contribution of that particular source corpus to every possible subset of the source corpora. Each block inside SEAL-Shap denotes a possible subset and the marginal contribution is derived by the difference of transfer results while trained with and without the corresponding source. Based on the source values, we select a subset of source corpora that achieves high transfer accuracy.

a good transfer performance but also for analyzing the source-target relationships.

Nonetheless, determining the value of a source corpus is challenging as it is affected by many factors, including the quality of the source data, the amount of the source data, and the difference between source and target at lexical, syntax and semantics levels (Ahmad et al., 2019; Lin et al., 2019). The current source valuation or ranking methods are often based on single source transfer performance (McDonald et al., 2011; Lin et al., 2019; Vu et al., 2020) or leave-one-out approaches (Tommasi and Caputo, 2009; Li et al., 2016; Feng et al., 2018; Rahimi et al., 2019). They do not consider the combinations of the sources. Consequently, they may identify the best single source corpus effectively but their top-k ranked source corpora may achieve limited gain in transfer results.

In this paper, we introduce *SEAL-Shap (Source*

sElection for trAnsfer Learning via Shapley value), a source valuation framework[2] (see Fig 1) based on the Shapley value (Shapley, 1952; Roth, 1988) in cooperative game theory. SEAL-Shap adopts the notion of Shapely value to understand the contribution of each source by computing the approximate average marginal contribution of that particular source to every possible subset of the sources.

Shapley value is a unique contribution distribution scheme that satisfies the necessary conditions for data valuation like fairness and additivity (Dubey, 1975; Jia et al., 2019a,b). As many model explanation methods including Shapley value are computationally costly (Van den Broeck et al., 2021), in a different context of features and data valuation in machine learning, Ghorbani and Zou (2019) propose to use an approximate Shapley value to estimate the feature or data values.

However, the existing approximation methods for estimating Shapley values are not scalable for NLP applications. NLP models are often large (e.g., BERT (Devlin et al., 2019)) and NLP transfer learning usually assumes a large amount of source data. To deal with the scalability issue, we propose a new sampling scheme, a truncation method, and a caching mechanism to efficiently approximate the source Shapley values.

We evaluate the effectiveness of SEAL-Shap under various applications in quantifying the usefulness of the source corpora and in selecting potential transfer sources. We consider two settings of source valuation or selection: (1) where a small target corpus is available; and (2) where we only have access to the linguistic or statistical features of the target, such as language distance to the sources, typological properties, lexical overlap etc. For the first setting, we use the small target data as the validation set to measure the values of the sources w.r.t the target. For the second setting, we follow Lin et al. (2019) to train a source ranker based on SEAL-Shap and the available features.

We conduct extensive experiments in both (zero-shot) cross-lingual and cross-domain transfer settings on three NLP tasks, including POS tagging, sentiment analysis, and natural language inference (NLI) with different model architectures (BERT and BiLSTM). In a case study, on the cross-lingual transfer learning, we exhibit that the source language values are correlated with the language

family and language distance—indicating that our source values are meaningful and follow the intuitive source-target relationships. Lastly, we analyze the approximation correctness and the run-time improvement of our source valuation framework SEAL-Shap.

2   Source Valuation Framework

We propose SEAL-Shap, a source valuation framework. We start with the setting where we have only one target and multiple sources. We denote the target corpus by V and the corresponding set of source corpora by $\mathcal{D} = \{D_1, \cdots, D_m\}$. Our goal is to quantify the value Φ_j of each source corpus D_j to the transfer performance on V and explain model behaviors. Once the source values are measured, we can then develop a method to select either all the sources or a subset of sources (i.e., $\subseteq \mathcal{D}$) that realizes a good transfer accuracy on V. Below, we first review the data Shapley value and its adaptation for transfer learning. Then, we describe how SEAL-Shap efficiently quantifies Φ_j and how to use it to select a subset of sources for model transfer.

2.1   Background: Data Shapley Value

Shapley value is designed to measure individual contributions in collaborative game theory and has been adapted for data valuation in machine learning (Ghorbani and Zou, 2019; Jia et al., 2019a,b). In the transfer learning setting, on a target corpus V, let $Score(C_\Omega, V)$ represent the transfer performance of a model C trained on a set of source corpora Ω.[3] The Shapley value Φ_j is defined as the average marginal contribution of a source corpus D_j to every possible subsets of corpora $\mathcal{D}$:

$$\frac{1}{m} \sum_{\Omega \subseteq \mathcal{D} - D_j} \frac{Score(C_{\Omega \cup D_j}, V) - Score(C_\Omega, V)}{\binom{m-1}{|\Omega|}}.$$

TMC-Shap for Transfer Learning: Computing the exact source-corpus Shapley value, described above, is computationally difficult as it involves evaluating the performances of the transfer models trained on all the possible combinations of the source corpora. Hence, Ghorbani and Zou (2019) propose to approximate the evaluation by

a truncated Monte Carlo method. Given the target corpus V and a set of source corpora $\mathcal{D}$, for each epoch, a source training data set $\Omega \subseteq \mathcal{D}$ is maintained and a random permutation π on $\mathcal{D}$ is performed (corresponds to line 6 in Algorithm 1 which is discussed in Sec 2.2). Then it loops over every source corpus π_j in the ordered list π and compute its marginal contribution by evaluating how much the performance improves by adding π_j to Ω: $Score(C_{\Omega \cup \pi_j}, V) - Score(C_\Omega, V)$. These processes are repeated multiple rounds and the average of all marginal contributions associated with a particular source corpus is taken as its approximate Shapley value (line 18 in Algorithm 1). When the size of Ω increase, the marginal contribution of adding a new source corpus becomes smaller. Therefore, to reduce the computation, Ghorbani and Zou (2019) propose to truncate the computations at each epoch when the marginal contribution of adding a new source π_j is smaller than a user defined threshold *Tolerance* (line 10-11, 18 in Algorithm 1).[4]

2.2 SEAL-Shap

Despite that TMC-Shap improves the running time, it is still unrealistic to use it in our setting where both source data and model are large. For example, in cross-lingual POS tagging on Universal Dependencies Treebanks, on average, it takes more than 200 hours to estimate the values of 30 source languages with multi-lingual BERT (See Sec 4.4). Therefore, in the following, we propose three techniques to further speed-up the evaluation process.

Stratified Sampling When computing the marginal contributions, training a model C on the entire training set Ω is computationally expensive. Based on extensive experiments, when computing these marginal contributions, we find that we do not need the performance difference of models trained with the entire training sets. For a reasonably large source corpus, 20-30% samples[5] in each source achieve lower but representative performance difference, in general. Therefore, we sample a subset of instances to evaluate the marginal contributions. To address computational limitation and scale to large data, sampling techniques have been widely discussed (L'heureux et al., 2017). In particular, we employ a stratified sampling (Neyman, 1992) to generate a subset $\mathcal{T}$

Algorithm 1: SEAL-Shap

Input: Source corpora $\mathcal{D} = \{D_1, \cdots, D_m\}$, target corpus V, Random sampler $\mathcal{S}$, sample size η, num of epochs $nepoch$, and Classifier C

Output: Source-corpora Shapley values $\{\Phi_1 ..., \Phi_m\}$

1 **Initialize:** Score cache $S \leftarrow \{\}$, source Shapley values $\Phi_x \leftarrow 0$ for $x = 1 \ldots m$, and epoch $t \leftarrow 0$

2 $\mathcal{D}_{samp} \leftarrow \{\mathcal{S}(D_x, \eta), \forall D_x \in \mathcal{D}\}$

3 $C_{\mathcal{D}_{samp}} \leftarrow$ Train C on $\mathcal{D}_{samp}$

4 **while** *Converge or $t < nepoch$* **do**

5     $t \leftarrow t + 1$

6   π : Random permutation of $\mathcal{D}$

7     $v_0 \leftarrow \rho$

8   **for** $j \in \{1, \cdots m\}$ **do**

9        $\Omega \leftarrow \{\pi_1, \cdots, \pi_j\}$

10   **if** $| Score(C_{\mathcal{D}_{samp}}, V) - v_{j-1}| < Tolerance$ **then**

11           $v_j \leftarrow v_{j-1}$

12   **else**

13   **if** $\Omega \notin S$ **then**

14              $\mathcal{T} \leftarrow \{\mathcal{S}(\Omega_x, \eta), \forall \Omega_x \in \Omega\}$

15              $C_j \leftarrow$ Train C on $\mathcal{T}$

16   Insert Ω into S with $S_\Omega \leftarrow$ $Score(C_j, V)$

17           $v_j \leftarrow S_\Omega$

18        $\Phi_{\pi_j} \leftarrow \frac{t-1}{t}\Phi_{\pi_j} + \frac{1}{t}(v_j - v_{j-1})$

---

from Ω by sampling training instances from each source corpus Ω_x with a user defined sample rate η. Then, we train the model on $\mathcal{T}$ (line 14-15 in Algorithm 1). The quantitative effectiveness of this technique is discussed in Sec 4.4 and the impact of different sampling rates are presented in Fig 5.

Truncation As discussed in Sec 2.1, at each epoch, Ghorbani and Zou (2019) truncate the computations once a marginal contribution becomes small when looping over the ordered list π of that corresponding epoch, typically for the last few sources in π. On the other hand, at the beginning of each epoch, when computing the marginal contribution by adding the first source corpus π_1 into an empty Ω, the contribution is computed by the performance gap between a model trained on π_1 and a random baseline model without any training. Usually, the performance of a random model (v_0) is low and hence, the marginal contribution is high in the first step, in general. As this scale of marginal contributions at the first step is drastically different from later steps, it leads TMC-Shap to converge slowly. Hence, to restrict the variance of the marginal contributions, we down weight the marginal contributions of the first step by setting $v_0 = \rho$, where ρ is a hyper-parameter[6] indicating the

---

[4]Setting *Tolerance* to 0 turns off the truncation.

[5]Higher sampling rate typically leads to better approximation but are expensive in run-time.

[6]Typically a factor of the performance achieved when using only one source, or all the sources together

baseline performance of a model (line 7, 18 in Algorithm 1).

Caching When computing the source Shapley values, we have to repeatedly evaluate the performance of the model on different subsets of source corpora. Sometimes, we may encounter subsets that we have evaluated before. For example, consider a set of source corpora $\mathcal{D} = \{D_1, D_2, D_3\}$ and we evaluate their Shapley values through two permutations: $\pi_1 = [D_3, D_1, D_2]$, and $\pi_2 = [D_1, D_3, D_2]$. When we compute the marginal contribution of the last source corpus D_2, in both cases the training set $\Omega = \{D_1, D_3\}$. That is, if we cache the result of $Score(C_{D_1 \cup D_3})$, then we can reuse the scores. We implement this cache mechanism in line 1, 13, 16, 17 in Algorithm 1. With these optimization techniques, we improve the computation time by about 2x (see Sec 4.4). This enables us to apply this techniques in NLP transfer learning.

Note that whenever an Ω causes a cache miss, for each source Ω_x, as discussed above in this Section, we sample a new set of instances (line 13-14 in Algorithm-1). Thus, given a reasonably large number of epochs, our approach performs sampling for a large number of times and in aggregation, it evaluates a wide number of samples in each source.

2.3 SEAL-Shap for Multiple Targets

Many applications require to evaluate the values of a set of sources with respect to a set of targets. For example, under the zero-shot transfer learning setting, we assume a model is purely trained on the source corpora without using any target data. Consequently, then the same trained model can be evaluated on multiple target corpora. With this intuition, whenever the model is trained on a new training set Ω, SEAL-Shap evaluates it on all the target corpora and caches all of them accordingly.

2.4 Source Values without Evaluation Corpus

In the previous discussions above, we assume a small annotated target corpus is available and can be used to evaluate the transfer performances. However, in some scenarios, only some linguistic or statistical features of the sources and targets, such as language distance and word overlap, are available. Lin et al. (2019) show that by using these features, we can train a ranker to sort the sources to unknown targets by predicting their value. In the following, we extend their ranker by incorporating it with SEAL-Shap.

Given the set of training corpora $\mathcal{D}$ and the actual target corpus V, we iteratively consider each training corpus D_j as target and the rest m-1 corpora as the sources. We compute the corresponding source values $\mathcal{Y}_{\mathcal{D}}^{D_j} = \{\Phi_{D_1}, \ldots, \Phi_{D_{j-1}}, \Phi_{D_{j+1}}, \ldots, \Phi_{D_m}\}$. Now, w.r.t the target D_j, the linguistic or statistical features of the source corpora (e.g., language distance from the target, lexical overlap between the corresponding source and the target) $\mathcal{X}_{\mathcal{D}}^{D_j} = \{F^j(D_1), \ldots, F^j(D_{j-1}), F^j(D_{j+1}), \ldots, F^j(D_m)\}$ where F^j denotes the source feature generator function for the corresponding target D_j. This feature vector of the source corpora ($\mathcal{X}_{\mathcal{D}}^{D_j}$) is a training input and their value vector ($\mathcal{Y}_{\mathcal{D}}^{D_j}$) is the corresponding training output for the ranker. We repeat this for each training corpus and generate the respective training inputs and outputs for the ranker. Once trained, for the actual target V and the source corpora $\mathcal{D}$, the ranker can predict the values of the source corpora $\mathcal{Y}_{\mathcal{D}}^V$ only based on the linguistic source features $\mathcal{X}_{\mathcal{D}}^V$.

2.5 Source Corpora Selection by SEAL-Shap

The source values computed in Sec 2.2-2.4 estimate the usefulness of the corresponding transfer sources and can be used to identify the potential sources which lead to the good transfer performances. We select the potential source corpora in two ways. (i) Top-k: We simply sort the sources based on their values and select the user defined top-k sources. (ii) Threshold: When an annotated evaluation dataset in target corpus V is available, after computing the source values, we empirically set a threshold θ and select each source that has source value higher than θ. On that evaluation target corpus, we tune and set θ for which the corresponding transfer model achieves the best performance.

3 Experimental Settings

We conduct experiments on *zero-shot cross-lingual and cross-domain transfer* settings. Models are trained only on the source languages/domains and directly applied in target languages/domains. **Cross-lingual Datasets** We conduct experiments on two popular cross-lingual transfer problems: (i) universal POS tagging on the Universal Dependencies Treebanks (Nivre et al., 2018). Following Ahmad et al. (2019), we select 31 languages of 13 different language families (details in Appendix A). (ii) natural language inference on the XNLI

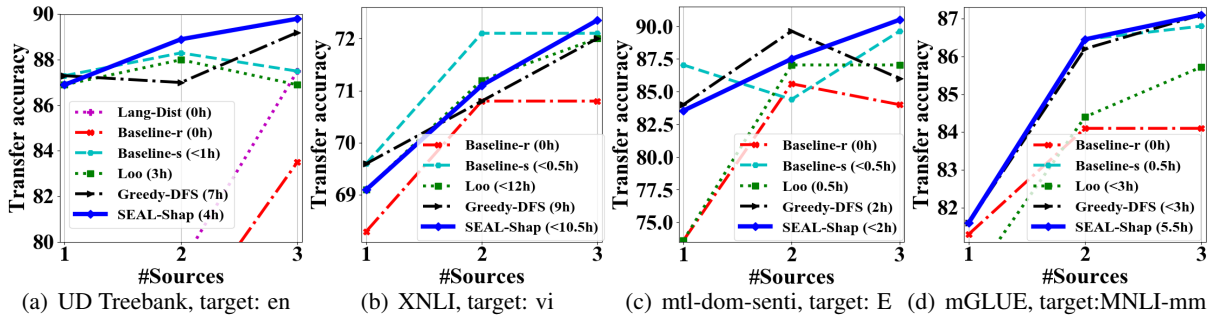| (a) UD Treebank, target: en | (b) XNLI, target: vi | (c) mtl-dom-senti, target: E | (d) mGLUE, target:MNLI-mm |

Figure 2: Performance, and run time with up to top-3 sources ranked by different approaches. (a), (b) denotes cross-lingual and (c), (d) denotes cross-domain transfer. All models have same training configurations (e.g., sample size). All the run times are final except for *Greedy DFS* where it increases linearly with top-k. Adding top-2 and top-3 ranked sources, other methods drop their accuracy across the tasks while ours shows a consistent gain in all tasks and achieves the best results with top-3 sources.

dataset (Conneau et al., 2018), that covers 15 different languages. XNLI task is a 3-way classification task (entailment, neutral, and contradiction). Data statistics are in Appendix R.

Cross-domain Datasets We consider three domain transfer tasks: (i) POS tagging: we use the SANCL 2012 shared task datasets (Petrov and McDonald, 2012) that has six different domains (details in Appendix B). (ii) Sentiment analysis: we use the multi-domain sentiment datasets (Liu et al., 2017) which has several additional domains than the popular Blitzer et al. (2007) dataset, See Appendix D. (iii) NLI: we consider a (modified) binary classification (e.g., entailed or not) dataset used in Ma et al. (2019). It is made upon modification on GLUE tasks (Wang et al., 2018) and has four domains (details in Appendix C). As GLUE test sets are unavailable, for each target domain, we use the original dev set as the pseudo test set and randomly select 2,000 instances from its training set as the pseudo dev set.

Classifier and Preprocessing For all domain transfer tasks, we use BERT and for all language transfer tasks, we use multi-lingual BERT (Devlin et al., 2019) models except for cross-doman POS tagging where we consider the state-of-the-art BiLSTM based Flair framework (Akbik et al., 2018). For BERT models, we use the Transformers implementations in the Huggingface library Wolf et al. (2019). For significance test, we use an open-sourced library.[7] By default, no preprocessing is performed except tokenization (see Appendix J).

Hyper-parameters Tuning For all BERT models, we tune the learning rate, batch size, and number of epochs. We also tune the number of epochs $nepoch$ in Algorithm 1, the threshold SEAL-Shap value θ, initial score ρ. Details are in Appendix K.

4 Results and Discussion

In the following, we first verify SEAL-Shap is an effective tool for source valuation. Then, we evaluate the source values when an evaluation target corpus is unavailable. In Sec 4.3, we interpret the relations between sources and targets based on the SEAL-Shap values. Finally, we analyze our method with comprehensive ablation studies.

4.1 Evaluating Source Valuation

We assess our source valuation approach in compare to the following baselines: (i) *Baseline-s*: source values are based on the single source transfer performance. (ii) *Leave-one-out (LOO)*: source values are based on how much transfer performance we loose if we train the model on all the sources except the corresponding one. (iii) *Baseline-r*: a random baseline that assigns random values to sources.[8] (iv) *Greedy DFS*: the top-1 ranked source is same as that of *Baseline-s*. Next, it selects one of the remaining sources as top-2 that gives the best transfer result along with the top-1 and so on. (v) *Lang-Dist*: (if available) in reverse order of target-source language distance (Ahmad et al., 2019).[9]

Balancing Source Corpora In the experiements, our focus is to understand the values of the sources. For some datasets, the sizes of source corpora are very different. For example, in UD Treebank, the number of instances in Czech, and Turkish is 69k, 3.5k, respectively. Since data-size is an obvious factor, we conduct experiments on balanced data to reduce the influence of data-size in the analysis. We sub-sample the source corpora to ensure their sizes are similar. Specifically, for the cross-domain NLI task, we sample 20k instances for each source.

---

[7]`github.com/neubig/util-scripts/blob/master/paired-bootstrap.py`

[8]Our experiments with different seeds result in different but similar results.

[9]Ahmad et al. (2019) compute the distances from an annotated dependency parse tree based on UD Treebank.

Lang	en	All Source	Baseline-r	Baseline-s	SEAL-Shap
en	-	82.71	86.32	86.39	**88.55***$†
fr	-	94.60	94.63	**94.83**	94.79
da	88.3	88.94	89.30	89.23	**89.47***
es	85.2	93.15	93.00	93.04	**93.21**$
it	84.7	96.58	96.43	**96.71**	96.67
ca	-	91.54	91.64	90.78	**92.08***$†
sl	84.2	93.28	93.50	92.89	**93.52***†
nl	75.9	90.10	90.19	90.14	**90.26**
ru	-	92.98	92.91	92.71	**93.13***$†
de	89.8	90.79	91.07	**91.44**	91.06
he	-	76.67	75.75	75.43	**76.73**$†
cs	-	93.89	93.04	93.94	**94.81***$†
sk	83.6	95.68	95.62	95.53	**95.81**†
sr	-	97.55	97.47	97.43	**97.58**†
id	-	84.10	85.23	85.50	**85.97***$
fi	-	**87.13**	86.89	86.86	87.05
ko	-	63.59	**64.27**	63.77	64.19
hi	-	81.49	80.27	79.94	**82.41***$†
ja	-	66.86	65.99	67.71	**67.81***$
fa	72.8	81.03	80.69	**82.37**	81.79
Average	-	82.98	83.05	83.15	**83.66**

Table 1: Performance on universal POS tagging when using each of language as the target language and the rest as source languages . '*', '$', '†' denote SEAL-Shap model is statistically significantly outperforms *All Sources*, *Baseline-r* and *Baseline-s* respectively using paired bootstrap test with p ≤ 0.05. "en" refers to the only source ("en") results in Wu and Dredze (2019).

For others, we sub-sample each source such that the size of the corpus is the same as the smallest one in the dataset. However, our approach can handle both balanced or unbalanced data and the source values are similar in conclusions (e.g., see Fig 5).
Result: We first compare these methods by selecting top-k sources ranked by each of the approach and reporting the corresponding transfer performance. With $k = 3$, we plot the corresponding transfer results and the running time for valuation in Fig 2. As mentioned in Sec 1, the relatively strong *Baseline-s* can select the best performing top-1 source but with top-2 and top-3 sources, the performances drop on cross-domain sentiment analysis and cross-lingual POS tagging (See Fig 2(c) and 2(a)) while our approach shows a consistent gain in all of the these tasks and with top-3 sources it achieves the best performances. Appendix I plots the results with higher k.

Next, as in Sec 2.5, we tune a threshold $θ$ and either select all the sources as useful or a smaller subset of m number of sources (i.e., $m < |\mathcal{D}|$) whose SEAL-Shap values are higher than $θ$. In the followings, we compare the model performances of these m sources selected by SEAL-Shap with the same top-m sources ranked by the aforementioned baseline methods. Being relatively weak or slow, we do not further report performances for *LOO,*

Model	WSJ	EM	N	A	R	WB	Avg
MMD	96.12	96.23	96.40	95.75	95.51	96.95	96.16
RENYI	96.35	96.31	96.62	95.52	95.97	96.75	96.25
All Sources	95.95	95.39	96.94	95.15	96.08	97.10	96.10
Baseline-r	95.98	93.41	93.78	93.14	95.25	97.10	94.78
SEAL-Shap	**96.14***$	**95.47***$	**97.02**$	**95.30***$	**96.17**$	97.10	**96.20**

Table 2: POS tagging results (% accuracy) on SANCL 2012 Shared Task. '*' and '$' denote the model using SEAL-Shap statistically significantly outperforms *All Sources* and *Baseline-r* respectively using paired bootstrap test with p ≤ 0.05. MMD, and RENYI refer to Liu et al. (2019a) which use auxiliary unlabelled data in the target domain and focus on instance selection. *Baseline-s* has exactly same results as SEAL-Shap.

Model	bg	ru	tr	ar	vi	hi	sw	ur	Avg
XLM-MLM	74.0	73.1	67.8	68.5	71.2	65.7	64.6	63.4	68.54
mBERT(en)	68.9	69.0	61.6	64.9	69.5	60.0	50.4	58.0	62.79
All Sources	74.03	73.59	65.21	68.94	74.39	67.31	52.67	64.37	67.56
Baseline-r	74.69	**74.53**	**65.85**	68.68	75.03	66.69	**52.97**	63.69	67.77
Baseline-s	73.23	73.73	65.67	68.36	74.11	67.07	52.59	63.31	67.26
Ours	**74.95**	73.85	65.63	**69.24**	**75.71**	**67.78**	52.73	**64.67**	**68.07**

Table 3: XNLI results. As a reference, we include two results from the recently published papers mBERT (Wu and Dredze, 2019) and "XLM-MLM" Lample and Conneau (2019). mBERT is trained on "en" only and "XLM-MLM" is applicable to XNLI languages only.

Lang-Dist, and *Greedy DFS*. Rather we consider another strong baseline *All Sources* that uses all the source corpora $\mathcal{D}$. This is a strong baseline as it is trained on more source-corpus instances in general.
Cross-Lingual POS Tagging We evaluate the source selection results on zero-shot cross-lingual POS tagging in Table 1. Among the 31 target languages, in 21 of them, SEAL-Shap selects a small subset of source corpora. From the Table, overall, SEAL-Shap selects source corpora with high usefulness for training the model, and except for few cases the model constantly outperforms all the baselines by more than 0.5% in avg token accuracy. In 13 of them, it is statistically significant by a paired bootstrap test. The gap is especially high for English, Czech, and Hindi. These results demonstrate that SEAL-Shap is capable in both quantifying the source values and also in source selection. We report the full results on the dev and test set of all target languages in Appendix M, N respectively. For each row in Table 1, the number of selected sources are reported in Appendix S.
Cross-Domain POS Tagging Table 2 presents the POS tagging results in zero-shot domain transfer on SANCL 2012 shared task. In 5 out of 6 targets, SEAL-Shap outperforms all baselines except *Baseline-s*. For each target domain with only 5

Model	books	kitchen	dvd	baby	MR	Avg
Cai and Wan (2019)	87.3	88.3	88.8	90.3	76.3	86.2
All Sources	**87.3**	90.3	88.3	92.3	79.3	87.5
Baseline-r	87.0	90.5	87.3	91.8	78.8	87.1
Baseline-s	86.8	89.8	87.0	**92.5**	77.5	86.7
SEAL-Shap	**87.3**	90.8	88.8	92.5	79.5	87.8

Table 4: Cross-domain transfer results on multi-domain sentiment analysis task. Cai and Wan (2019) use unlabelled data from the target domain.

Model	SNLI	QQP	QNLI	MNLI-mm	Avg
Ma et al. (2019)	88.30	73.90	59.10	-	76.23
All Sources	88.69	72.96	50.65	89.47	75.45
Baseline-r	88.11	72.71	50.53	89.18	75.13
Baseline-s	**88.72**	**73.47**	50.98	**89.69**	75.72
SEAL-Shap	**88.72**	**73.47**	**54.75**	**89.69**	**76.66**

Table 5: Zero-shot results on modified GLUE. Ma et al. (2019) selects instances from one source domain at once while we select a subset of source corpora.

sources, *Baseline-s* source values match with ours in general. However, SEAL-Shap significantly outperforms *Baseline-r* on all 5 cases and *All-Sources* twice. It even outperforms MMD, and RENYI (Liu et al., 2019a) on Newsgroups (N), Reviews (R), and Weblogs (WB) despite they select source data at instance level and use additional resources.

Cross-Lingual NLI In Table 3, we show the XNLI results in 8 target languages where SEAL-Shap selects a small subset of source corpora. Among them, in 3 languages, *Baseline-r* marginally surpasses ours. However, in 5 other languages SEAL-Shap outperforms all the baselines with clear margin specially on Bulgarian, Vietnamese with about 1% better accuracy (full results in Appendix E).

Cross-Domain NLI Next, we evaluate SEAL-Shap on the modified GLUE dataset in Table 5. SEAL-Shap outperforms *Baseline-s* once and other baselines in all cases. Its highest performance improvement is gained on QNLI, where it outperforms others by 4%.

Cross-Domain Sentiment Analysis Among the 13 target domains in the multi-domain sentiment analysis dataset, in 5 domains SEAL-Shap selects a small subset (full results in Appendix O). As in Table 4), with a large margin, SEAL-Shap achieves higher accuracy than all other baselines and, in 4 cases, it is even better than Cai and Wan (2019) that uses unlabeled target data.

Our experimental evidences show that SEAL-Shap is an effective tool in choosing useful transfer sources and can achieve higher transfer performances than other source valuation approaches.
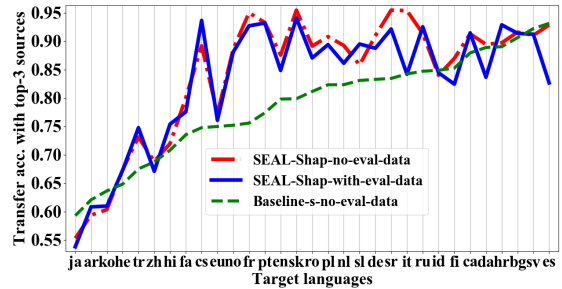

Figure 3: Cross-lingual POS tagging accuracies on different target languages using top-3 sources ranked by SEAL-Shap. The ranker (red) selects similar sources as using SEAL-Shap with annotated target data (blue). Ranker trained to predict SEAL-Shap values (red) performs better than baseline (green) (Lin et al., 2019).
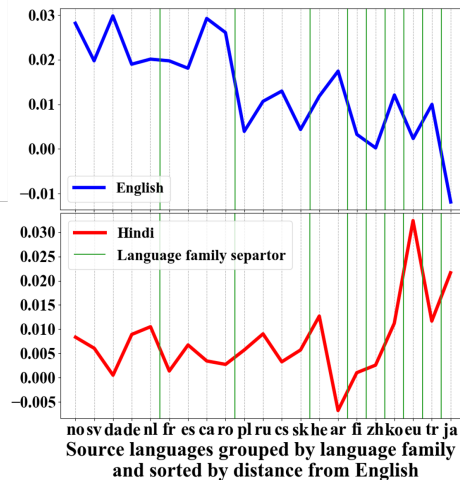

Figure 4: Cross-lingual POS tagging SEAL-Shap values, referring to the relative contribution of the source languages.

4.2 Results without an Evaluation Corpus

We evaluate the effectiveness of SEAL-Shap to build a straightforward ranker that directly computes the source values without any evaluation target corpus (see Sec 2.4). We use the ranker in Lin et al. (2019) as the underlying ranking model. First, we show that the source values evaluated by the ranker is as good as SEAL-Shap that uses its annotated target dataset. We compare the transfer performances of the top-k sources based on the source values computed with and without the evaluation corpus. Then, we show that the ranker trained with SEAL-Shap is more effective than training it with the existing single source based *Baseline-s*.

In cross-lingual POS tagging on UD Treebank, for each of the 31 target languages, we set aside that language and consider the remaining 30 languages as the training corpora. We then train the ranker as described in Sec 2.4 and compute the source values using it. As for reference, we pass the evaluation target dataset and the 30 source languages to SEAL-Shap to compute their values on

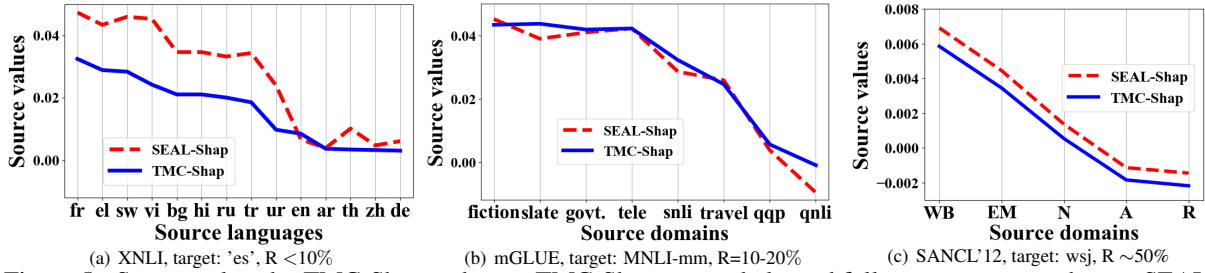(a) XNLI, target: 'es', R <10% (b) mGLUE, target: MNLI-mm, R=10-20% (c) SANCL'12, target: wsj, R ∼50%

Figure 5: Source values by TMC-Shap and ours. TMC-Shap uses unbalanced full source corpora whereas SEAL-Shap that achieves similar source values uses balanced and sampled source corpora. Even with a small sample rate (R), source order is almost same. Higher sampling rate typically refers to better approximation but leads to expensive runtime. In general, for a reasonably large corpus, 20-30% samples (>few thousands) are found sufficient to achieve reasonable approximation.

the evaluation dataset. With $k = 3$, we compare the transfer results of the top-k sources of these two methods in Fig 3. We also plot the results of the baseline ranker (Lin et al., 2019) that is trained with *Baseline-s*. Results show that the ranker source values are similar to the sources values estimated by SEAL-Shap with an annotated evaluation dataset and also it outperforms the baseline.

4.3 Interpret Source Value by SEAL-Shap

In this Section, we show that SEAL-Shap values provide a means to understand the usefulness of the transfer sources in cross-lingual and cross-domain transfer. We first analyze cross-lingual POS tagging. Following Ahmad et al. (2019), we consider using language family and word-order distance as a reference distance metric. We anticipate that languages in the same language family with smaller word-order distance from the target language are more valuable in multi-lingual transfer. We plot SEAL-Shap of source languages evaluated on two target languages English ("en") and Hindi ("hi") in Fig 4. In the x-axis, a common set of twenty different source languages are grouped into ten different language families and sorted based on the word order distance from English. As the figure illustrates, Germanic and Romance languages have higher Shapley values when using English as the target language. The value gradually decreases for language of other families when the word order distance increase. As for the target language Hindi, the trend is opposite, in general.

Figure 6: Similar SEAL-Shap value curves for two closely related target languages in cross-lingual POS tagging.
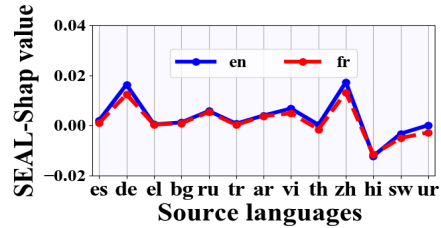

Figure 7: Similar SEAL-Shap value curves for two closely related XNLI targets "en" and "fr". In XNLI, the source corpora are prepared by machine translating from "en". This data processing may affect the source values. Translation into "zh" being relatively better, although different from both targets, its source values are higher than others.

Analogously, in cross-domain NLI, we find that correlation between QNLI, and QQP is high whereas between MNLI-mm and QQP, it is lower (see Appendix Q).

SEAL-Shap on Similar Targets Intuitively, if two target corpora are similar, the corresponding Shapley values of the source corpora when transferring to these two targets should be similar as well. To verify, in Fig 6, we plot the Shapley values of twenty nine source languages for targets Russian and Serbian on cross-lingual POS tagging. Also we plot the source values when transferring a NLI model to English and French in Fig 7. We observe that the corresponding curves are almost identical, and SEAL-Shap in fact selects the same set of source corpora as potential. These results suggest that if there is no sufficient data in the target corpus, it is also possible to use a neighboring corpus as a proxy to compute SEAL-Shap values.

Source Values Influenced by Data Processing Typically, the sources with least or negative source values are from the domains/languages that are different from the targets (e.g., Fig 4). However, in some cases, source usefulness (i.e., values) is affected by the data preparing process. For example, in XLNI, the source corpora are prepared by machine translation from "en" (Conneau et al., 2018) and the quality of this translation into "zh" is better

Prob.	Transfer	Target	#Targets	#Samples	Caching	Time (hours)
NLI	Domain	MNLI-mm	1	✗	✗	300*
			1	✗	✓	101
			1	20k	✓	18
			3	20k	✓	5
POS	Language	Arabic (ar)	1	✗	✗	210*
			1	✗	✓	180*
			1	3.3k	✓	25
			31	3.3k	✓	3.5

Table 6: Running time for computing approximate Shapley value. The marker *represents the time is estimated by extrapolation. #Targets indicates number of target corpus evaluated simultaneously. #Samples is the number of samples used to train model for computing marginal contribution. TMC-Shap is equivalent to disable all the techniques (the first row of each block).
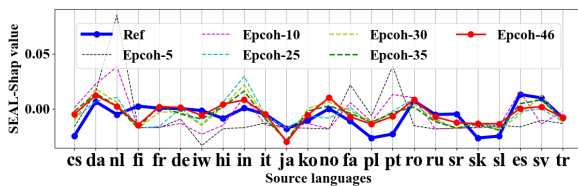

Figure 8: SEAL-Shap value with two (colored) seeds.

in compare to other languages, in general. Consequently, in Fig 7, "zh" has higher source value for both targets "en" and "fr".

4.4 Analysis and Ablation Study

Finally, we analyze the proposed Algorithm 1 for computing Shapley value approximately.

How good is the approximation? In Fig 5, we compare SEAL-Shap with TMC-Shap (Ghorbani and Zou, 2019) on three datasets (details in Appendix F). Overall, the Shapley values obtained by SEAL-Shap and TMC-Shap are highly correlated and their relative orders are matched, while SEAL-Shap is much more efficient. Note that, the rankings themselves being same/similar, the model performances using the same/similar top-k sources are same/similar, too; therefore, we do not list their transfer performances furthermore.

Ablation Study: We examine the effectiveness of each proposed components in SEAL-Shap. Results are shown in Table 6 and details are in Appendix F-H. Results show that without the proposed approximation, TMC-Shap is computational costly and is impractical to use to analyze the value of source corpus in the NLP transfer setting. All the proposed components contribute to significantly speed-up the computations.

Is the approximation sensitive to the order of permutations? As SEAL-Shap is a Monte Carlo

approximation, we study if SEAL-Shap is sensitive to the random seed using the cross-lingual POS tagging task. To analyze, we first compute a reference Shapley values by running SEAL-Shap until empirically convergence (blue line). Then, we report the Shapley value produced by another random seed. Fig 8 shows that with enough epochs, the values computed by different random seeds are highly correlated (more in Appendix H).

5 Related Work

As discussed in Section 1, transfer learning has been extensively studied in NLP to improve model performance in low-resource domains and languages. In the litearture, various approaches have been proposed to various tasks, including text classification (Zhou et al., 2016; Kim et al., 2017), natural language inference (Lample et al., 2018; Artetxe and Schwenk, 2019), sequence tagging (Täckström et al., 2013; Agić et al., 2016; Kim et al., 2017; Ruder and Plank, 2017), dependency parsing (Guo et al., 2015; Meng et al., 2019). These prior studies mostly focus on bridging the domain gap between sources and targets.

In different contexts, methods including influence functions and Shapley values have been applied to value the contribution of training data (Koh and Liang, 2017; Lundberg et al., 2018; Jia et al., 2019a). Specifically, Monte Carlo approximation of Shapley values has been used in various applications (Maleki, 2015; Jia et al., 2019a; Ghorbani and Zou, 2020; Tripathi et al., 2020; Tang et al., 2020; Sundararajan and Najmi, 2019). However they are either task/model specific or not scalable to NLP applications. Oppositely, Kumar et al. (2020) discuss the problems of using Shapley value for model explanation. In contrast, we apply efficient Shapley value approximation in NLP transfer learning and analyze the source-target relationships.

6 Conclusion

We propose SEAL-Shap to quantify the value of the source corpora in transfer learning for NLP by computing an approximate Shapley value for each corpus. We show that SEAL-Shap can be used to select source corpora for transfer and provide insight on understanding the value of source corpora. In the future, we plan to further improve the runtime of our source valuation approach by limiting the repetition of model training.

7   Acknowledgments

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*, 4:301–312.

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

Yitao Cai and Xiaojun Wan. 2019. Multi-domain sentiment classification based on domain-aware embedding and attention. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4904–4910. AAAI Press.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Pradeep Dubey. 1975. On the uniqueness of the Shapley value. *International Journal of Game Theory*, 4(3):131–139.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728.

Amirata Ghorbani and James Zou. 2019. Data Shapley: Equitable valuation of data for machine learning. *International Conference on Machine Learning*, pages 2242–2251.

Amirata Ghorbani and James Zou. 2020. Neuron Shapley: Discovering the responsible neurons. *arXiv preprint arXiv:2002.09815*.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1234–1244.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas Spanos, and Dawn Song. 2019a. Efficient task-specific data valuation for nearest neighbor algorithms. *Proceedings of the VLDB Endowment*, 12(11):1610–1623.

Ruoxi Jia, Xuehui Sun, Jiacen Xu, Ce Zhang, Bo Li, and Dawn Song. 2019b. An empirical and comparative analysis of data valuation with scalable algorithms. *arXiv preprint arXiv:1911.07128*.

Joo-Kyung Kim, Young-Bum Kim, Ruhi Sarikaya, and Eric Fosler-Lussier. 2017. Cross-lingual transfer learning for pos tagging without cross-lingual resources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2832–2838.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*.

I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. 2020. Problems with Shapley-value-based explanations as feature importance measures. *ICML Workshop on Workshop on Human Interpretability in Machine Learning (WHI)*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3125–3135, Florence, Italy. Association for Computational Linguistics.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019a. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1957–1968.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019b. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. 2018. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.

Alexandra L'heureux, Katarina Grolinger, Hany F Elyamany, and Miriam AM Capretz. 2017. Machine learning with big data: Challenges and approaches. *Ieee Access*, 5:7776–7797.

Xiaofei Ma, Peng Xu, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2019. Domain adaptation with bert-based domain classification and data selection. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 76–83.

Sasan Maleki. 2015. Addressing the computational issues of the Shapley value with applications in the smart grid.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72.

Tao Meng, Nanyun Peng, and Kai-Wei Chang. 2019. Target language-aware constrained inference for cross-lingual dependency parsing. 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 1117–1128.

Jerzy Neyman. 1992. On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. In *Breakthroughs in Statistics*, pages 123–150. Springer.

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164.

Alvin E Roth. 1988. *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with Bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark. Association for Computational Linguistics.

Lloyd S Shapley. 1952. A value for n-person games. Technical report, Rand Corp Santa Monica CA.

Mukund Sundararajan and Amir Najmi. 2019. The many Shapley values for model explanation. *CoRR*, abs/1908.08474.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12.

Siyi Tang, Amirata Ghorbani, Rikiya Yamashita, Sameer Rehman, Jared A Dunnmon, James Zou, and Daniel L Rubin. 2020. Data valuation for medical imaging using Shapley value: Application on a large-scale chest x-ray dataset. *arXiv preprint arXiv:2010.08006*.

Tatiana Tommasi and Barbara Caputo. 2009. The more you know, the less you learn: from knowledge transfer to one-shot learning of object categories. In *British Machine Vision Conference*.

Sandhya Tripathi, N Hemachandra, and Prashant Trivedi. 2020. On feature interactions identified by Shapley values of binary classification games. *arXiv preprint arXiv:2001.03956*.

Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. 2021. On the tractability of shap explanations. *In Proceedings of the 35th Conference on Artificial Intelligence (AAAI)*.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. 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 833–844, Hong Kong, China. Association for Computational Linguistics.

Y. Yang, X. Li, P. Wang, Y. Xia, and Q. Ye. 2020. Multi-source transfer learning via ensemble approach for initial diagnosis of alzheimer's disease. *IEEE Journal of Translational Engineering in Health and Medicine*, 8:1–10.

Y. Yao and G. Doretto. 2010. Boosting for transfer learning with multiple sources. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1855–1862.

Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Shen-Shyang Ho. 2016. Transfer learning for cross-language text categorization through active correspondences construction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2400–2406.

Supplementary Material: Appendices

A Details of UD Treebanks

We use the Flair framework provided version of UDTreebank.

The statistics of the Universal Dependency treebanks (v2.2) is summarized in Table 7. However, more accurate statistics can be found using the above link.

Language	Lang. Family	Treebank	Num. of	Sent.	#Token(w/o punct)
Arabic (ar)	Afro-Asiatic	PADT	train	6075	223881(206041)
			dev	909	30239(27339)
			test	680	28264(26171)
Bulgarian (bg)	IE.Slavic	BTB	train	8907	124336(106813)
			dev	1115	16089(13822)
			test	1116	15724(13456)
Catalan (ca)	IE.Romance	AnCora	train	13123	417587(371981)
			dev	1709	56482(50452)
			test	1846	57902(51459)
Chinese (zh)	Sino-Tibetan	GSD	train	3997	98608(84988)
			dev	500	12663(10890)
			test	500	12012(10321)
Croatian (hr)	IE.Slavic	SET	train	6983	154055(135206)
			dev	849	19543(17211)
			test	1057	23446(20622)
Czech (cs)	IE.Slavic	PDT,CAC,CLTT,FicTree	train	102993	1806230(1542805)
			dev	11311	191679(163387)
			test	12203	205597(174771)
Danish (da)	IE.Germanic	DDT	train	4383	80378(69219)
			dev	564	10332(8951)
			test	565	10023(8573)
Dutch (nl)	IE.Germanic	Alpino,LassySmall	train	18058	261180(228902)
			dev	1394	22938(19645)
			test	1472	22622(19734)
English (en)	IE.Germanic	EWT	train	12543	204585(180303)
			dev	2002	25148(21995)
			test	2077	25096(21898)
Estonian (et)	Uralic	EDT	train	20827	287859(240496)
			dev	2633	37219(30937)
			test	2737	41273(34837)
Finnish (fi)	Uralic	TDT	train	12217	162621(138324)
			dev	1364	18290(15631)
			test	1555	21041(17908)
French (fr)	IE.Romance	GSD	train	14554	356638(316780)
			dev	1478	35768(31896)
			test	416	10020(8795)
German (de)	IE.Germanic	GSD	train	13814	263804(229338)
			dev	799	12486(10809)
			test	977	16498(14132)
Hebrew (he)	Afro-Asiatic	HTB	train	5241	137680(122122)
			dev	484	11408(10050)
			test	491	12281(10895)
Hindi (hi)	IE.Indic	HDTB	train	13304	281057(262389)
			dev	1659	35217(32850)
			test	1684	35430(33010)
Indonesian (id)	Austronesian	GSD	train	4477	97531(82617)
			dev	559	12612(10634)
			test	557	11780(10026)
Italian (it)	IE.Romance	ISDT	train	13121	276019(244632)
			dev	564	11908(10490)
			test	482	10417(9237)
Japanese (ja)	Japanese	GSD	train	7164	161900(144045)
			dev	511	11556(10326)
			test	557	12615(11258)
Korean (ko)	Korean	GSD,Kaist	train	27410	353133(312481)
			dev	3016	37236(32770)
			test	3276	40043(35286)
Norwegian (no)	IE.Germanic	Bokmaal,Nynorsk	train	29870	489217(432597)
			dev	4300	67619(59784)

			test	3450	54739(48588)
Polish (pl)	IE.Slavic	LFG,SZ	train	19874	167251(136504)
			dev	2772	23367(19144)
			test	2827	23920(19590)
Portuguese (pt)	IE.Romance	Bosque,GSD	train	17993	462494(400343)
			dev	1770	42980(37244)
			test	1681	41697(36100)
Romanian (ro)	IE.Romance	RRT	train	8043	185113(161429)
			dev	752	17074(14851)
			test	729	16324(14241)
Russian (ru)	IE.Slavic	SynTagRus	train	48814	870474(711647)
			dev	6584	118487(95740)
			test	6491	117329(95799)
Serbian (sr)	IE.Slavic	SET	train	3328	74259(74259)
			dev	599	11993(11993)
			test	600	11421(11421)
Slovak (sk)	IE.Slavic	SNK	train	8483	80575(65042)
			dev	1060	12440(10641)
			test	1061	13028(11208)
Slovenian (sl)	IE.Slavic	SSJ, SST	train	8556	132003(116730)
			dev	734	14063(12271)
			test	1898	24092(22017)
Spanish (es)	IE.Romance	GSD,AnCora	train	28492	827053(730062)
			dev	3054	89487(78951)
			test	2147	64617(56973)
Swedish (sv)	IE.Germanic	Talbanken	train	4303	66645(59268)
			dev	504	9797(8825)
			test	1219	20377(18272)
Turkish (tr)	Altaic	IMST	train	36822	37784(36822)
			dev	988	10046(9777)
			test	983	10029(9797)
Basque (eu)	Language Iasolate	BDT	train	5396	72974(72974)
			dev	1798	24095(24095)
			test	1799	24074(24374)
Persian (fa)	IE.Iranic	UPDT	train	4798	121064(119945)
			dev	599	15832(15755)
			test	600	16020(15925)

Table 7: Statistics of the UD Treebanks we used. For language family, "IE" stands for Indo-European and "(w/o) punct" means #tokens excluding "PUNCT" and "SYM".

Language Family	Languages
Afro-Asiatic	Arabic (ar), Hebrew (he)
Austronesian	Indonesian (id)
IE.Germanic	Norwegian (no), Danish (da), Dutch (nl), English (en), German (de), Swedish (sv)
IE.Indic	Hindi (hi)
IE.Altaic	Turkish (tr)
IE.Romance	Catalan (ca), French (fr), Portuguese (pt), Italian (it), Romanian (ro), Spanish (es)
IE.Slavic	Bulgarian (bg), Croatian (hr), Czech (cs), Polish (pl), Russian (ru), Slovak (sk), Slovenian (sl), Serbian (sr)
Japanese	Japanese (ja)
Korean	Korean (ko)
Sino-Tibetan	Chinese (zh)
Uralic	Finnish (fi)
Iranic	Persian (fa)
Isolate	Basque (eu)

Table 8: **The selected languages from UDTreebank 2.2, used in our cross lingual POS tagging, grouped by language families. "IE" is the abbreviation of Indo-European.**

B POS Taggong Dataset for Domain Transfer

Domain	WSJ	Emails	Newsgroups	Answers	Reviews	Weblogs
Train/Dev/Test	2976/1336/1640	4900/2450/2450	2391/1196/1195	3489/1745/1744	3813/1907/1906	2031/1016/1015

Table 9: **Data Statistics of SANCL 2012 shared task dataset (Petrov and McDonald, 2012)**

C Modified GLUE NLI Task for Domain Transfer

For NLI, we consider the 2-class classification (e.g., entailed or not) corpora used in Ma et al. (2019) that is made upon modification on 4 Glue benchmark (Wang et al., 2018) problems: SNLI, MNLI, QNLI, and QQP. We split the MNLI training set into a corpora of "fiction", "slate", "govt.", "travel", and "telephone" as in Williams et al. (2018) and always include them in the source corpora for all target domains. Here, As the annotations for GLUE test sets are publicly unavailable, for each target domain, we consider the original dev set as pseudo test set and randomly select 2k instances from training set for parameter tuning (i.e., pseudo dev set). For MNLI as target, we have two original dev set. Hence, we take the 2k instances from matched dev set as pseudo dev set and consider the miss-matched corpus as pseudo test set. Therefore, in zero-shot setting, the number of source corpora for target MNLI is 8, and for others it is 7.

Task Category	Dataset	Train Size	Dev Size
Natural Language Inference	SNLI	510,711	-
	MNLI-Fiction	77348	-
	MNLI-Travel	77350	-
Multi-Genre	MNLI-Slate	77306	-
Natural Language Inference	MNLI-Government	77350	-
	MNLI-Telephone	83348	-
	MNLI-Mismatched	-	9,832
Answer Sentence Selection	QNLI	108,436	5,732
Paraphrase Detection	QQP	363,847	-

Table 10: **Data Statistics of Glue NLI tasks. We report the performance on the full dev set and to tune all the models, randomly select 20% examples from it.**

D Sentiment Analysis Dataset for Domain Transfer

For sentiment analysis, following Cai and Wan (2019), we use the multi-domain sentiment datasets released by Liu et al. (2017) which has several additional domains than a popular sentiment analysis dataset Blitzer et al. (2007). For each domain, we use the same test set as in Liu et al. (2017). However, as train and dev data are released together, we simply consider the first section of this combined set as the train set and the last section as dev set. Statistics of the 14 domains in this dataset considered in our experiments are reported in Appendix.

Dataset	Train	Dev	Test	Avg length
Books	1400	200	400	159
Electronics	1398	200	400	101
DVD	1400	200	400	173
Kitchen	1400	200	400	89
Apparel	1400	200	400	57
Camera	1397	200	400	130
Health	1400	200	400	81
Toys	1400	200	400	90
Video	1400	200	400	156
Baby	1300	200	400	104
Magazine	1370	200	400	117
Software	1315	200	400	129
Sports	1400	200	400	94
MR	1400	200	400	21

Table 11: **Data Statistics of multi-domain sentiment dataset**

E XNLI Results

Model	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	en	fr
All Sources	77.88	71.82	72.23	74.03	73.59	65.21	68.94	74.39	60.10	74.69	67.31	52.67	64.37	82.65	77.03
Baseline-r	77.88	71.82	72.23	74.69	**74.53**	**65.85**	68.68	75.03	60.10	74.69	66.69	**52.97**	63.69	82.65	77.03
Baseline-s	77.88	71.82	72.23	73.23	73.73	65.67	68.36	74.11	60.10	74.69	67.07	52.59	63.31	82.65	77.03
Ours	77.88	71.82	72.23	**74.95**	73.85	65.63	**69.24**	**75.71**	60.10	74.69	**67.78**	52.73	**64.67**	82.65	77.03

Table 12: **Cross-lingual results on the XNLI test sets.**

All model performances are same when selecting all source corpora as potential.

F How good is the approximation?

We consider three different datasets: XNLI (target: 'es'), Modified GLUE NLI dataset(target: 'MNLI-mm'), and SANCL 2012 shared task for POS tagging (target: 'WSJ'). We use the corresponding full size source tasks except for the extremely large XNLI in which we randomly sample half of each source task (180k instances) and compute the Shapley value adopting the source code released by Ghorbani and Zou (2019). Then, on XNLI dataset we consider sample size 50k, on GLUE 20k, on SANCL 2012 sahred task 2k for each source task. Then we use Algorithm 1 (in the main paper) with tuned initial score to compute the approximate data Shapley value. Instead of full convergence, we do early stop by setting the Shapley value nepochs to 10, 50, 30 on XNLI, GLUE, and SANCL datsets respectively.

G   Shapley Value Computation Time with/out different Factors:

We consider two example problem to transfer both language and domain: (i) UDPOS tagging for language transfer (ii) modified GLUE NLI for domain transfer. We consider the "initial score" to *All Sources/2* and $\mathcal{R}$; $nepoch$ to 30, and 50 for these two respective target task, for the data Shapley computation as in Algorithm 1, we then switch different factors as in reported in Table 6 (in the main paper) as record the corresponding Shapley value computation time.

H Approximate Shapley Value with Different Seeds:

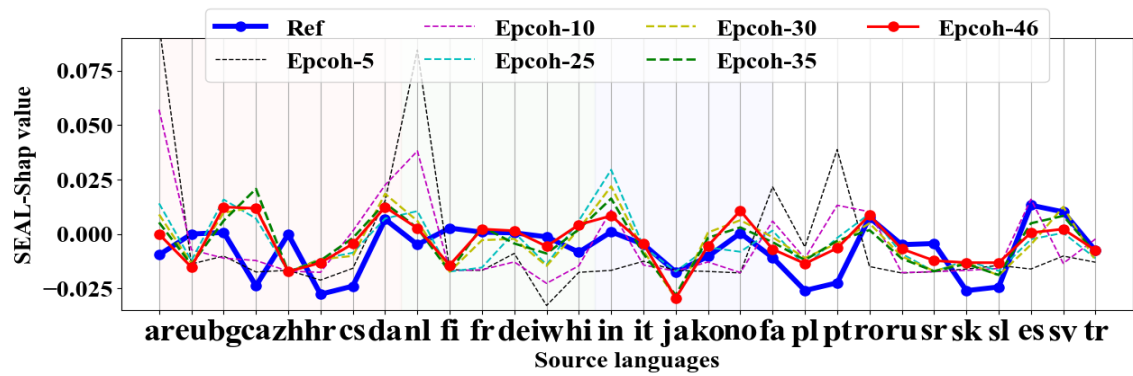In Figure 9, we plot the SEAL-Shap value w.r.t same threshold ($\theta_k = 0$) by different seeds.

Figure 9: SEAL-Shap value with two different seeds.

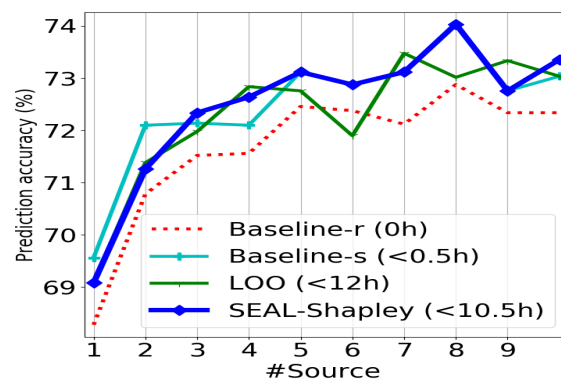# I Adding sources according different source ranking/selection methods

Figure 10: Performance using top-k sources as per dfifferent source ranking/selection methods. (Task: XNLI, target: vi. Red colored line denotes Random or Baseline-S.
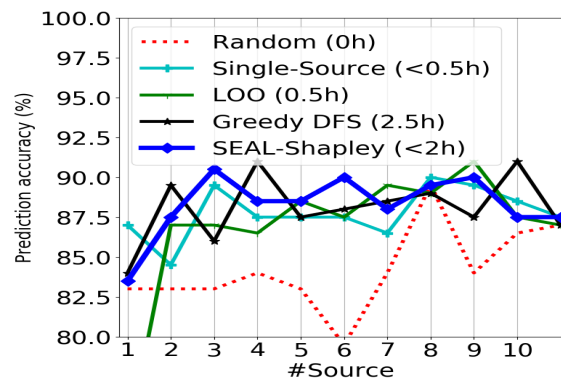

Figure 11: Performance using top-k sources as per different source ranking/selection methods (Task: Cross-domain Sentiment Analysis, Target: Electronics.) Red colored line denotes Random or Baseline-S.
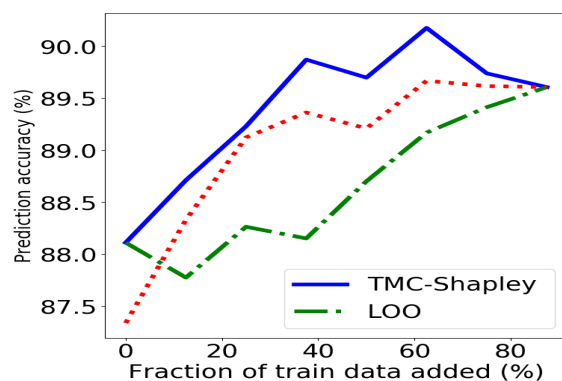
Figure 12: Performance using top-k sources as per dfifferent source ranking/selection methods. (Task: Cross-domain NLI, target: MNLI-mm). Red colored line denotes Random or Baseline-S.

We consider a Bert model with certain model parameters. Then using the corresponding training and development dataset we compute SEAL-Shap values by adjusting the ρ, compute the ranks according to Baseline-s, Baseline-r, language distance from the target language etc., We consider the top-3 sources to compare. We also consider the top-3 sources in a greedy depth first search approach. Ours get consistent increase and best performance using top-3 sources. Here we plot figure with more top-3 sources. For large datasets, we do not plot the greedy DFS here as it takes extremely long time to compute due to the fact that the DFS search branches rarely overlap for different targets.

J Classifier and Data Preprocessing:

As for the underlying machine learning classifier, in our experiments on the domain transfer problem, we consider the BERT based cased model (Devlin et al., 2019) except for POS tagging. For POS tagging, we consider the the state-of-the-art BiLSTM based Flair framework (Akbik et al., 2018). As for language transfer problem, we consider a generic state-of-the-art classifier: the multi-linigual version of BERT based cased model. For all bert models, we adopt Transformers implementation (Wolf et al., 2019). Number of model parameters BERT model 10 million parameters. For each task, no preprocessing is performed other than the tokenization of words into subwords with WordPiece except for cross-lingual POS for which we use an oppen-sourced multilingual preprocessing toolkit[10] to remove "strange control character" tokens. Following Wu and Dredze (2019), we also limit subwords sequence length to 128 to fit in a single GPU for all tasks. For all tasks, we use the accuracy metric.

---

[10] github.com/huggingface/transformers/tree/master/examples/token-classification

K Hyper-parameters Tuning:

For the small multi-domain sentiments analysis dataset, we do a full search of the combination of learning rate, batch size and number epochs up to 5. For all other large scale datasets, we perform a greedy search. We first find the best combination of learning rate and batch size. Then we tune the number of epochs. For the extremely large XNLI, in which for any target task the multi-source training data size is $\sim 5.5M$, we tune only when our framework select a smaller subset of the source corpora for learning rate in $\{3\times10^{-5}, 5\times10^{-5}\}$, batch size $\{32\}$ and epochs within 50k steps (i.e., no more than 3 epochs). On XNLI, when our framework selects all source corpora as potential, we do not further tune the hyper-parameters both baselines have the same training set as SEAL-Shap. Hence, we report the result using a default learning rate 5^{-5}, batch size 32. All test results reported in this paper are performed on the corresponding test set[11] using a single gpu. All Shapley value calculations were performed on multi-gpus. After transfer source selection, all models for XLNI, and UDPOS, are trained, and tuned on multi-gpu distributed system and for for SANCL 2012 POS tagging, and mulit-domain sentiment analysis datasets single gpu is used. As for modified GLUE NLI dataset, both single gpu and distributed system is used. For UDPOS significance test, we use the default num_samples 10k, except for Polish we use 3k. As for Flair, the system does not support saving the prediction options, getting the model prediction even from a trained BiLSTM model is time consuming. Hence, we sample for no more than 50 times for SANCL 2012 dataset significance test. For Flair framework, after preliminary verification, we follow their configuration suggested for best performance on English Penn treebank POS tags[12] and tune each model up to 150 epochs with patience 4. All the approximated Shapley values (SEAL-Shap values) are computed within $nepoch$ 30 and only for UDPOS and XNLI dataset, multiple seeds (< 3) are used. For UDPOS $nepoch$ within 30 or 46, for XNLI $nepoch$ within 10 or 20. For any target task V_k, the corresponding threshold Shapley value θ_k is chosen in $\{1\times10^{-2}, 1\times10^{-3}, 5\times10^{-3}\}$, initial scores ρ in $\{\mathcal{R}, \mathcal{N}, 0.5, \text{ All sources}/2, \text{ All Sources}, \mu\}$ where $\mathcal{R}$ is a random baseline model performance (i.e., randomly initialized model performance); given the total number of sources n, $\mathcal{N} = \frac{n-1}{n} \times Score(C_{D_j}, V_k)$; $\mu = mean(\{\text{ All Sources}\} \cup \{Score(C_{D_j}, V_k) \forall D_j \in D\})$, and D is all source tasks. This means we also tune SEAL-Shap value as the mean of a combination of SEAL-Shap values, leave one out values, and single source transfer values. For Shapley value computation, for multiple seeds run like on cross-lingual POS tagging and XNLI, seed 42, 43 is used. All the hyper-parameter tuning is done with default seed in the open-sourced Transformer implementation[13] which is 42. All the SEAL-Shap values are calculated using single seed. Only for plotting Figure 6 in main paper, on cross-lingual POS tagging for target English two different seeds are used. The blue curve in Figure 6, and the results in Table 2 are using the same seed and all other plots uses the other seed. All the parameter configuration and the dev set performance will be reported here upon acceptance. All computations are performed on gpus; in general using (4,8,1) #gpus. Note that while tuning, if there is no θ for which the corresponding subset of sources (i.e., $\subset \mathcal{D}$) achieves better result than using all of $\mathcal{D}$, then we select the set of all sources $\mathcal{D}$ assuming each source is contributing positively.

---

[11]for GLUE NLI, pseudo test set
[12]github.com/flairNLP/flair/blob/master/resources/docs/EXPERIMENTS.md
[13]github.com/huggingface/transformers/

L Training a Direct Source Selection Ranker using SEAL-Shap
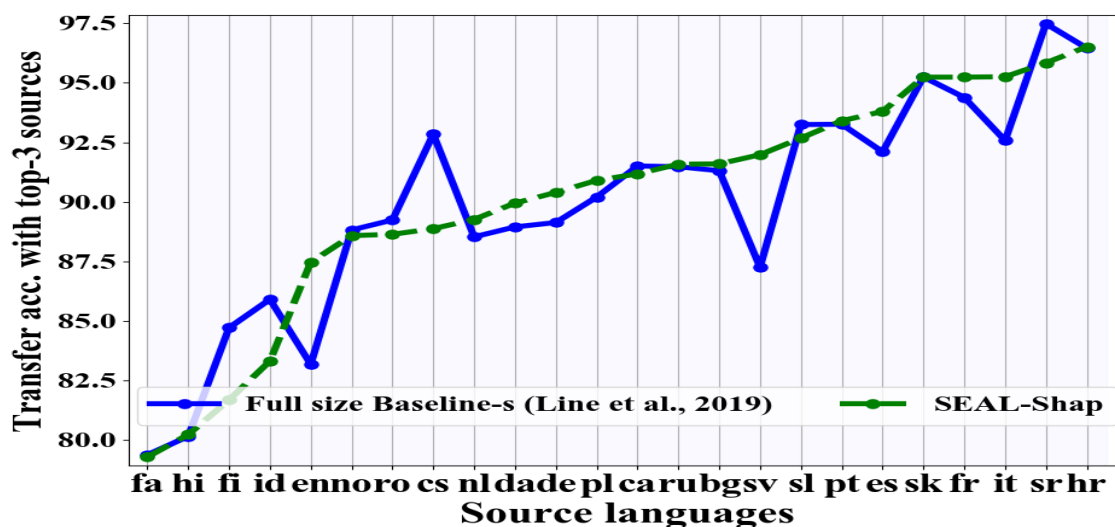
Figure 13: Transfer performance with the top-3 sources

In addition to the sampled single source performances *Baseline-s* in the main paper, here we also compare smapled SEAL-Shap with the full size (i.e., no sampling) single source performance. Lin et al. (2019) with full size *Baseline-s* results are found using the original ranker realised in (Lin et al., 2019). In most cases ours outperforms (Lin et al., 2019) such as 'hr', 'de', 'da', 'nl', 'en', 'fr', 'he', 'it', 'es', 'sv'. However, the margin is small and also there are multiple cases where Lin et al. (2019) outperforms ours such as 'ar', 'cs', 'zh', 'id', 'fi', 'ja', 'ko', 'sr'.

M Dev set Results

In Table 13, we report the dev set result for cross-lingual POS tagging. For French, German, Hebrew, Slovenian, we need

Lang	All Sources	Baseline-s	SEAL-Shap
en	85.21	87.39	**88.50**
no	90.05	90.05	90.05
sv	**93.39**	93.27	93.18
fr	95.52	95.68	**95.71**
pt	94.55	94.73	**94.77**
da	90.19	90.27	**90.42**
es	94.11	94.04	**94.16**
it	96.83	96.56	**96.89**
hr	96.36	96.36	96.36
ca	92.58	92.39	**92.92**
pl	**91.64**	91.40	91.62
sl	93.35	93.56	**93.45**
nl	91.47	91.55	**91.55**
bg	92.26	92.26	92.26
ru	92.87	92.79	**92.92**
de	91.42	91.65	91.42
he	77.09	76.16	**77.30**
cs	94.56	93.14	**94.74**
ro	90.41	90.41	90.41
sk	96.38	96.33	**96.42**
sr	97.18	97.27	**97.35**
id	83.98	84.63	**85.58**
fi	87.24	87.26	87.26
zh	71.31	71.31	71.31
ar	79.18	79.18	79.18
ko	63.58	63.76	**64.31**
hi	80.69	80.16	**82.78**
ja	69.28	69.72	**69.93**
tr	78.43	78.43	78.43
eu	80.90	80.90	80.90
fa	82.37	81.67	**82.74**

Table 13: Dev set results on cross-lingual POS tagging.

N Full Cross-lingual POS Tagging Results

Lang	en	All Source	Baseline-r	SEAL-Shap	Baseline-s
en	-	82.71	86.32	**88.55***$	86.39
no	-	90.06	90.06	90.06	90.06
sv	83.6	93.26	93.26	93.26	93.26
fr	-	94.60	94.63	94.79	**94.83**
pt	82.1	94.33	94.33	94.33	94.33
da	88.3	88.94	89.30	**89.47***	89.23
es	85.2	93.15	93.00	**93.21**$	93.04
it	84.7	96.58	96.43	96.67	**96.71**
hr	-	96.60	96.60	96.60	96.60
ca	-	91.54	91.64	**92.08***$	90.78
pl	86.9	91.61	91.61	91.61	91.61
sl	84.2	93.28	93.50	**93.52***	92.89
nl	75.9	90.10	90.19	**90.26**	90.14
bg	87.4	92.93	92.93	92.93	92.93
ru	-	92.98	92.91	**93.13***$	92.71
de	89.8	90.79	91.07	91.06	**91.44**
he	-	76.67	75.75	**76.73**$	75.43
cs	-	93.89	93.04	**94.81***$	93.94
ro	84.7	89.97	89.97	89.97	89.97
sk	83.6	95.68	95.62	**95.81**	95.53
sr	-	97.55	97.47	**97.58**	97.43
id	-	84.10	85.23	**85.97***$	85.50
fi	-	**87.13**	86.89	87.05	86.86
zh	-	71.31	71.31	71.31	71.31
ar	-	80.07	80.07	80.07	80.07
ko	-	63.59	**64.27**	64.19	63.77
hi	-	81.49	80.27	**82.41***$	79.94
ja	-	66.86	65.99	**67.81***$	67.71
tr	-	78.43	78.43	78.43	78.43
eu	-	81.18	81.18	81.18	81.18
fa	72.8	81.03	80.69	81.79	**82.37**
Average	-	87.17	87.21	**87.62**	

Table 14: Performance on universal POS tagging (test set) when using each of language as the target language and the rest as source languages . '*' and '$' denote SEAL-Shap model is statistically significantly outperforms *All Sources* and *Baseline-s* respectively using paired bootstrap test with p ≤ 0.05. 'en' refers to the best single source ('en') results, reported in Wu and Dredze (2019).

All model performances are same when selecting all source corpora as potential. (See line 2 in Table 13 and Table 14).

O   Full Cross-domain Sentiment Analysis Results

Model	books	kitchen	dvd	electronics	apparel	camera	baby	health	magazines	MR	software	video	toys	sports	Avg
Cai and Wan (2019)	87.3	88.3	88.8	89.5	88.0	90.3	90.3	91.0	88.5	76.3	90.8	91.3	90.3	90.5	82.16
All Sources	**87.3**	90.3	88.3	90.8	91.0	91.5	92.3	92.0	90.5	79.3	90.3	85.3	91.3	90.5	89.33
Baseline-r	87.0	90.5	87.3	90.8	91.0	91.5	91.8	92.0	90.5	78.8	90.0	84.8	91.3	90.5	89.08
Baseline-s	86.8	89.8	87.0	90.8	91.0	91.5	**92.5**	92.0	90.5	77.5	90.0	84.8	91.3	90.5	-
SEAL-Shap	**87.3**	**90.8**	**88.8**	90.8	91.0	91.5	**92.5**	92.0	90.5	**79.5**	90.3	87.8	91.3	90.5	**89.76**

Table 15: Cross-domain Transfer performance on multi-domain sentiment analysis dataset (Liu et al., 2017). Cai and Wan (2019) leverages unlabelled data from the target domain.
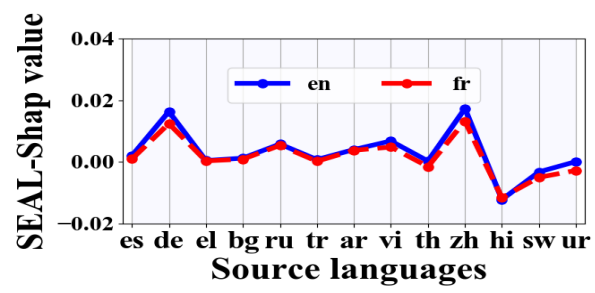
P SEAL-Shap values for two similar targets

Figure 14: Similar SEAL-Shap value curvature of two close language English ("en") and French ("fr") on cross-lingual NLI.

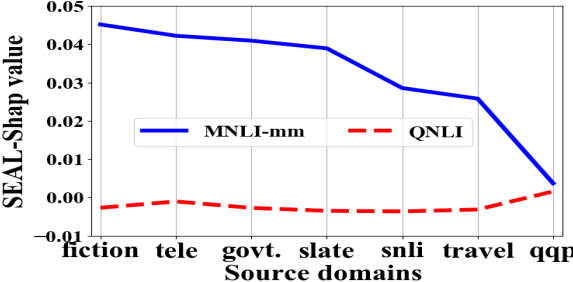## Q   Interpreting Source Shapley Values in Cross-domain NLI

Figure 15: SEAL-Shap value on cross-domain NLI, referring to relative contribution of source domains. For target domain MNLI-mm, source domain QQP has the lowest contribution, whereas for target domain QNLI, source domain QQP has the highest contribution.

R Data Statistics

Transfer	Task	Dataset	#target	#source
Language	POS tag	UD Treebank	31	30
	NLI	XNLI	15	14
Domain	POS tag	SANCL 2012	6	5
	NLI	mGLUE	4	7+
	Sentiment Ana.	mlt-dom-senti	14	13

Table 16: Task statistics. #sources are for each target. In (m)odified GLUE, #sources is 8 for target MNLI, and 7 otherwise. "mlt-dom-senti" refers to Liu et al. (2017).

S Number of Sources Selected

Lang	#Sources Selected
en	9
fr	29
da	29
es	27
it	26
ca	25
sl	29
nl	28
ru	27
de	28
he	29
cs	27
sk	27
sr	27
id	26
fi	27
ar	30
ko	27
hi	27
ja	29
fa	27

Table 17: Number of sources selected from 30 different languages by SEAL-Shap for the task of cross-lingual POS tagging.