# Using Transfer Learning to Assist Exploratory Corpus Annotation

Paul Felt, Eric K. Ringger, Kevin D. Seppi, Kristian Heal<sup>†</sup>

Brigham Young University

Department of Computer Science, <sup>†</sup>Neal A. Maxwell Institute

Provo, UT 84602 USA

paul\_felt@byu.edu, {ringger, kseppi}@cs.byu.edu, kristian\_heal@byu.edu

#### Abstract

We describe an under-studied problem in language resource management: that of providing automatic assistance to annotators working in exploratory settings. When no satisfactory tagset already exists, such as in under-resourced or undocumented languages, it must be developed iteratively while annotating data. This process naturally gives rise to a sequence of datasets, each annotated differently. We argue that this problem is best regarded as a transfer learning problem with multiple source tasks. Using part-of-speech tagging data with simulated exploratory tagsets, we demonstrate that even simple transfer learning techniques can significantly improve the quality of pre-annotations in an exploratory annotation.

Keywords: corpus annotation, transfer learning, machine learning

# **1.** Exploratory Corpus Annotation (ECA)

Because corpora are useful for investigating the structure of language, studying the way that languages change over time, testing linguistic hypotheses, charting the movement of ideas and historical trends, and even improving the effectiveness of language teaching and acquisition, they are an essential linguistic resource (Kroch, 1989; Sinclair, 2004; Nesselhauf, 2004). One of the most urgent needs for annotated corpora is in the realm of under-resourced and endangered language documentation (Grenoble and Whaley, 1998; Crystal, 2002; Bird and Simons, 2003; Gippert et al., 2006).

In domains such as under-resourced language documentation, annotation is unavoidably exploratory and iterative in nature (Hovy and Lavid, 2010). The annotator proposes an annotation scheme, annotates data, and then revises that annotation scheme in light of insights generated by applying the annotation scheme to real world data (Figure 1.), a process which for brevity we refer to as ECA (exploratory corpus annotation). ECA results in a sequence of possibly disjoint annotation sets, or "versions",  $V_1 \oplus \ldots \oplus V_K = V$ , where each  $V_v$  consists of data and annotations,  $\{(x_i, y_i)\}_{i=1}^{N_v}$ , produced according to  $V_v$ 's annotation scheme.

Each time the annotation scheme changes, some cost is incurred as existing annotations are invalidated and must be updated before the corpus is complete. The cost associated with evolving annotation schemes is largely a hidden cost, since few annotation projects record or report internal changes. For example, the Natural Language Processing (NLP) Lab at BYU is collaborating with scholars of ancient languages at the Neal A. Maxwell Institute for Religious Scholarship to create a large corpus of annotated Classical Syriac.<sup>1</sup> Although significant time was spent at the outset defining the annotation scheme that would be used, as preliminary data has been annotated at least a dozen updates have already been made to the annotation scheme. Since

http://cpart.maxwellinstitute.byu.edu/ home/sec/ we are starting with a sizable body of already annotated text, some of these changes have required considerable time and effort to implement (via re-annotation).

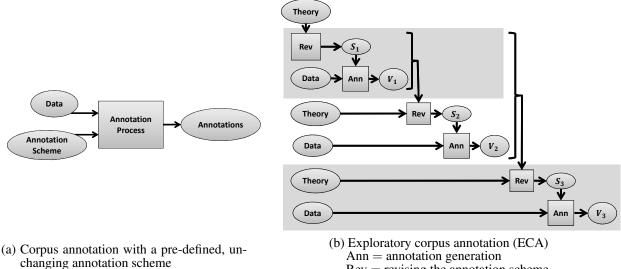
Annotation scheme revisions are especially likely in exploratory annotation scenarios dealing with languages or linguistic theories that have not previously been codified into annotation schemes. However, revisions can occur even in well established annotation tasks such as English part of speech tagging and parsing. When creating the Penn Treebank corpus, Marcus et al. (1993) re-annotated the Brown corpus data with revised part-of-speech tags. Additionally, Marcus et al. report that after publishing the Penn Treebank, they identified a variety of limitations and inconsistencies in their annotation scheme for English syntactic parsing and subsequently spent a good deal of effort repairing the parsing scheme and re-annotating data for future releases (Marcus et al., 1995).

A more extreme case comes from the SUSANNE corpus, another derivative of the Brown Corpus, annotated with very detailed parsing information. The 512-page book describing the SUSANNE annotation scheme required twelve years of work to finish, and the author describes the accompanying 130,000-word corpus as a "byproduct of the work of creating the SUSANNE annotation scheme" (Sampson, 2008). These examples underscore the effort involved in developing a satisfactory annotation scheme, even for mainstream languages and linguistic annotation tasks.

The costs involved in iteratively improving an annotation scheme mean that budget-constrained corpus developers often must choose between developing a linguistically optimal annotation scheme and generating useful amounts of annotated data. To make matters worse, statistical preannotation—the traditional method of reducing annotation overhead—is hampered by the lack of a self-consistent training set.

#### 2. Previous Work

Numerous annotation projects have shown that assisting annotators with good pre-annotations is essential to an-



Rev = revising the annotation scheme

Figure 1: Two Kinds of Corpus Annotation

notator efficiency and accuracy. Studies in English partof-speech tagging, Chinese parsing, information extraction, named entity recognition, and Semitic morphological analysis all demonstrate that high accuracy pre-annotations correlate strongly with low annotation cost (Marcus et al., 1993; Chiou et al., 2001; Culotta and McCallum, 2005; Ganchev et al., 2007; Felt et al., 2012). This point is critical to our future decision (see Section 4.) to focus on increasing model accuracy as a stand-in for reduced cost.

Because accurate pre-annotation models are so effective in reducing annotation effort, much work has been done to train high quality models with as little data as possible. The active learning literature aims to reduce the cost required to train a model by selecting instances for annotation that are likely to be most informative (Settles, 2010). Weakly supervised techniques attempt to speed model training by learning from unlabeled instances, or by allowing annotators to communicate their knowledge to the model by specifying labels or constraints that are applicable to large classes of data instances (Roth and Yih, 2004; Druck et al., 2009; Liang et al., 2009; Ganchev et al., 2010).

We know of no previous work that explicitly addresses the problem of providing automatic assistance for ECA; however, corpus developers have naturally gravitated towards the solution of adapting knowledge from the data in out-of-date versions. For example, the creators of the Penn Treebank corpus used an altered version of the Brown Corpus's annotation scheme, which can be seen as an example of a single large step in the iterative process of ECA (Marcus et al., 1993). Although the existing Brown Corpus annotations were unsuitable for direct use, the creators of the Penn Treebank used an automatic tagging model trained on heuristically modified Brown Corpus data to automatically pre-annotate Penn Treebank data. Although imperfect, these pre-annotations effectively doubled annotation speed, greatly reducing annotation cost (Francis and Kucera, 1979; Church, 1988).

#### Transfer Learning 2.1.

Using knowledge from one or more source tasks to improve performance on a target task, as the Penn Treebank developers did, is known as transfer learning, and is an area of active research within machine learning. Providing preannotations for ECA fits naturally into the transfer learning framework. The following definition of transfer learning borrows notation and ideas from Pan and Yang (2010), but with minor changes to highlight connections between transfer learning and the motivation presented in Section 1..

Definition 1 (Transfer Learning) Let D denote a domain comprising the feature space  $\mathcal{X}$  and a distribution p(x)over data  $x \in \mathcal{X}$ . Let  $\mathcal{T}$  denote a **task**, or annotation scheme, comprising a feature space  $\mathcal{X}$ , a label space  $\mathcal{Y}$ , and a labeling function  $f : x \to y$  where  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}^2$  Finally, let version  $V_t$  be the set of annotations produced according to the annotation scheme of task  $\mathcal{T}_t$ . Then the goal of transfer learning is to use data from all source versions  $V_{1..t-1}$  to improve our ability to model the target annotation scheme  $\mathcal{T}_t$ .

Definition 1 encompasses a large number of scenarios. There may be one or many source versions. Different quantities of data and annotations may be available in any given version. Furthermore, each version is associated with an annotator, domain, and task, and therefore may differ from other versions in terms of  $\mathcal{X}$ , p(x),  $\mathcal{Y}$ , or f. Each of these differences can be understood via simple examples. Text and images come from domains  $\mathcal{D}_1$ ,  $\mathcal{D}_2$  where  $\mathcal{X}_1 \neq \mathcal{X}_2$ . Poetry and newswire text come from domains where  $p_1(x) \neq p_2(x)$ . When two part-of-speech tagging tasks use different tagsets, then  $\mathcal{Y}_1 \neq \mathcal{Y}_2$ ; when they assign different meanings to the same tag,  $f_1 \neq f_2$ . Each setting

<sup>&</sup>lt;sup>2</sup>Although f may be approximately described in annotation manuals, the true f is generally unseen. In probabilistic approaches, f is often modeled as p(y|x)

of these variables in a versioned dataset  $V_{1...t}$  corresponds to a different transfer learning scenario.

Much work on transfer learning for NLP is currently in domain adaptation, the transfer setting in which  $\mathcal{T}_s = \mathcal{T}_t$ ,  $\mathcal{D}_s \neq \mathcal{D}_t$ , and the domains differ only in the marginal distribution of the data,  $p_s(X)$  and  $p_t(X)$ . An example of domain adaptation would be using Wikipedia text in which named entities (people, places, events, etc.) have been labeled in order to improve named entity recognition in movie reviews.

Other notable related work includes multi-task learning, a transfer setting in which  $\mathcal{D}_s = \mathcal{D}_t$  and there are multiple tasks that all differ from one another. Multi-task learning is unusual in that no source/destination distinctions made among tasks (Caruana, 1997). For example, Collobert et al. (2011) construct a system that simultaneously learns partof-speech tagging, named entity recognition, chunking, and semantic role labeling. Each task helps to inform the others, leading to higher performance on all tasks learned jointly than was possible for any individual task when learned individually.

# 3. ECA as Transfer Learning

We formally define the problem of providing machine assistance in the setting of exploratory corpus annotation as a transfer learning problem and introduce some simple solutions adapted from previous work. The solutions described below are appealing since they are conceptually simple and easy to implement using existing models as building blocks.

#### Definition 2 (Exploratory Corpus Annotation) The

transfer learning setting in which the following are true. There are multiple source tasks  $\mathcal{T}_{1..t-1}$ . For each pair i, j of source tasks,  $\mathcal{D}_i = \mathcal{D}_j$  and  $\mathcal{T}_i \neq \mathcal{T}_j$ . Finally, each source version has at least some labeled data. Little or no labeled data is available for the target version  $V_t$ .

Only a few of the possible transfer learning settings are commonly studied, and none of those match Definition 2. ECA is unusual and interesting from a transfer learning point of view because it has multiple source tasks and there is often a sequential relationship among the tasks.

### 3.1. Baselines: TGTTRAIN and ALLTRAIN

Let TGTTRAIN be the approach that ignores all data from source tasks and trains a traditional supervised classification model only on the current target data  $V_t$ . We can expect TGTTRAIN to do well when  $V_t$  is large and badly when  $V_t$  is small, such as at the beginning of a project or just after a change is made to the annotation scheme. TGT-TRAIN corresponds to annotation projects that discard outdated annotations when a new version is introduced. In practice, this tends to happen during the initial stages of a project, when the perceived value of the information being lost is low.

Let ALLTRAIN be the algorithm that trains a traditional supervised classification model on all available data  $V_{1...t}$ , ignoring version boundaries. We would expect ALL-TRAIN to do well when there are few differences between the source and target datasets, and badly when there are large differences.

# **3.2. STACK**

STACK refers to an adaptation of stacked generalization, in which traditional supervised models are trained on each of the datasets, and a higher level model is trained to accomplish the target task using the predictions of the lower level models as features (Wolpert, 1992). The higher level model can potentially discover patterns in the errors of the underlying models in order to know which are trustworthy in which contexts, and whether their guesses are wrong in ways that can be predictably mapped to the correct answer.

## 3.3. AUGMENT

AUGMENT is a simple and effective domain adaptation technique proposed by Daumé (2007). AUGMENT moves the data into a feature space that allows traditional supervised learning techniques to find commonalities and differences among data from different domains. It is assumed that there are two datasets: the source  $X_s$  and the target  $X_t$ . Each source feature vector is mapped into the new feature space by the kernel function  $\Phi^s(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x}, 0 \rangle$ , and each target feature vector is mapped by the function  $\Phi^t(\boldsymbol{x}) = \langle \boldsymbol{x}, 0, \boldsymbol{x} \rangle$ . Thus each feature has a source-specific version, a target-specific version, and a general version in the new feature space. This can be generalized to the context of multiple source domains by defining  $\Phi^{s1}(\boldsymbol{x}) = \langle \boldsymbol{x}, 0, 0, ..., \boldsymbol{x} \rangle$ ,  $\Phi^{s2}(\boldsymbol{x}) = \langle 0, \boldsymbol{x}, 0, ..., \boldsymbol{x} \rangle, ..., \Phi^t(\boldsymbol{x}) = \langle 0, 0, ..., \boldsymbol{x}, \boldsymbol{x} \rangle$ .

# 4. Experiments

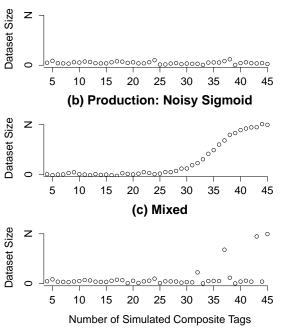
We would like to test the hypothesis that transfer learning can improve pre-annotations for ECA. However, evaluating a model in this setting requires access to corpora that recorded every version of the data V since the beginning of the project. We are aware of no such datasets. However, we can simulate such corpora by starting with an existing annotated corpus and probabilistically generating sequences of intermediate versions that explain how that corpus's annotation scheme might have come to be. For example, to start in familiar territory, we use the following process to create versioned datasets explaining the derivation of the Penn Treebank's part-of-speech tagged data.

0	<b>ithm 1</b> Simulate Versioned POS Datasets
	<b>:</b> GoldData is the reference dataset
Giver	<b>:</b> GoldTags is the reference tagset
1: <i>T</i>	$Cags \leftarrow CompositeTag(GoldTags)$
2: d	ataset $\leftarrow \{\}$
3: w	hile $Tags \neq $ GoldTags do
4:	$op \leftarrow sample(\{\text{SPLIT,MOVE,MERGE}\})$
5:	$Tags \leftarrow apply(op, Tags)$
6:	$\kappa \leftarrow sampleVersionSize()$
7:	$dataset \leftarrow annotate(\kappa, Tags, GoldData)$
8: re	eturn dataset

Algorithm 1 starts by grouping all the reference tags into a single composite tagset, then iterates between altering the tagset and annotating data until the original tagset is reached. A SPLIT represents an annotator deciding that the largest composite tag in the tagset is too broad and dividing it. A MERGE represents an annotator deciding that the distinctions between two tags are too fine and lumping them together. A MOVE represents an annotator moving one of the reference tags out of one composite tag and into another; in other words, deciding that a set of words that was previously labeled as something would be better labeled something else.

In order to be linguistically reasonable, splits are determined by finding a min-cut in the graph of reference tags, where reference tags are connected with strong adhoc weights if they are in the same family of tags (e.g., nouns, verbs, punctuation, etc), and weak weights otherwise. Merges and moves are chosen by sampling from a distribution over reference tag pairs where pairs that are identified by the Penn Treebank tagging guidelines as confusable (25 of these) or very confusable (9 of these) are more probable (Santorini, 1990).

Finally, sampleVersionSize() is implemented by hypothesizing that annotation projects alternate between small annotation batches for development and large batches for production approaching the size of the desired corpus, N, as the tagset converges on the reference set. The final mix favors development mode, since production mode involves heavy costs in the later stages (Figure 2).



(a) Development: Noisy Gaussian

Figure 2: Example draws from *sampleVersionSize()* 

This simulated data is clearly not ideal, and we and are actively working on developing real-world ECA data based on annotation projects we are involved in (Felt et al., In Press). However, in the meantime, simulated data allows us to make cautious observations about the characteristics of the problem and projections about the potential of transfer learning models to improve pre-annotation for ECA.

We used Algorithm 1 to generate 30 diverse datasets, choosing values for the simulation parameters at random.

We used maximum-entropy Markov models ("maxent taggers") with standard features (Toutanova et al., 2003) to implement the transfer algorithms described in Section 3.. Figure 3 shows the learning curve of each algorithm on a single dataset. TGTTRAIN's learning curve shows deep valleys at each version transition, because it is equivalent to beginning an entirely new learning curve at the beginning of each version. ALLTRAIN, on the other hand, shows a much smoother pattern. Using old data allows it to avoid the valleys seen in TGTTRAIN, but hurts its ability to reach high peaks quickly. STACK and AUGMENT both have peaks similar to TGTTRAIN, but manage to recover more quickly from version changes and avoid the low valleys.

	Heldout AAUC	Train Secs	Eval Secs
TGTTRAIN	225.4	0.2	0.003
ALLTRAIN	218.0	7.7	0.004
STACK	240.3	6.5	<u>0.046</u>
AUGMENT	246.4	<u>60.9</u>	0.006

Table 1: Bolded accuracies are significantly (p-val<0.01) better than non-bolded competitors. Underlined times are significantly (p-val;0.01) worse than non-underlined competitors. AAUC is averaged over 30 datasets. "Train Secs" means model training time averaged over all datasets. "Eval Secs" means average seconds to infer the labeling of a single sentence.

Because we are interested in a model that performs well at all stages of the ECA process, we need to compare entire learning curves rather than just final accuracy. A natural summary statistic for the quality of a learning curve is average area under the curve (AAUC). An accurate estimate of AAUC requires good resolution on the learning curve, so we evaluate between 500 and 1000 points on each learning curve, sampling more densely around version transitions. In Table 1 we report the average AAUC of each algorithm over all the simulated datasets, along with model timing statistics.

STACK and AUGMENT significantly outperform both TGTTRAIN and ALLTRAIN. Notice that AUGMENT is particularly slow to train, since its feature space has been expanded linearly in K, the number of versions. STACK is unusually slow at inference time, since it must solicit predictions from K subordinate models as features. The fact that TGTTRAIN was relatively easy to beat is encouraging, suggesting that there is room for more sophisticated transfer learning to make significant improvements over the baseline approach.

#### 5. Conclusions and Future Work

We have described the problem of providing automatic assistance to annotators working in exploratory settings. We have argued that this problem should be regarded as a transfer learning problem, and shown that existing transfer learning techniques can be adapted to significantly improve the quality of pre-annotations in simulated exploratory partof-speech tagging. Corpus annotators working in novel an-

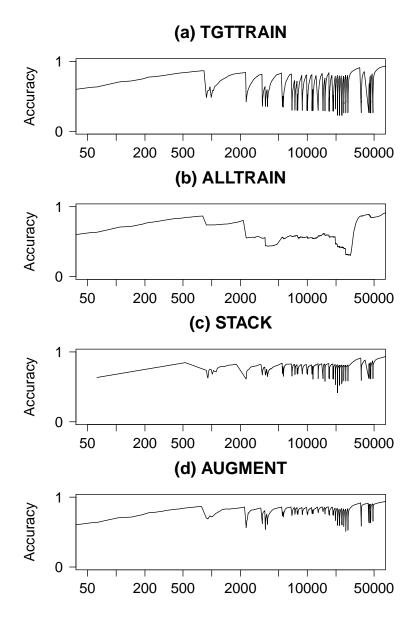


Figure 3: An example learning curve for each algorithm on the same dataset.

notation domains should be encouraged by these results and by the existence of a rich body of transfer learning work to draw on.

We plan to develop models that leverage the sequential nature of the versions. We also plan to apply the insights developed in this paper to improve pre-annotations for annotators engaged in real-world annotation projects. Finally, in order to apply these techniques seamlessly in annotation projects, it would be beneficial to discover a way of learning to automatically identify the boundaries between versions so that annotators need not manually identify annotation scheme changes.

# 6. References

- Bird, S. and Simons, G. (2003). Seven Dimensions of Portability for Language Documentation and Description. *Language*, pages 557–582.
- Caruana, R. (1997). Multitask learning. *Machine learn-ing*, 28(1):41–75.

- Chiou, F.-D., Chiang, D., and Palmer, M. (2001). Facilitating Treebank Annotation Using a Statistical Parser. In Proceedings of the First International Conference on Human Language Technology Research, pages 1–4.
- Church, K. W. (1988). A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In Proceedings of the Second Conference on Applied Natural Language Processing, pages 136–143. Association for Computational Linguistics.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.
- Crystal, D. (2002). *Language Death*. Cambridge University Press.
- Culotta, A. and McCallum, A. (2005). Reducing Labeling Effort for Structured Prediction Tasks. In *Proceedings* of the 20th Conference on Artificial Intelligence, pages 746–751. AAAI.

- Daumé III, H. (2007). Frustratingly easy domain adaptation. In Annual Meeting of the Association for Computational Linguistics, volume 45, page 256.
- Druck, G., Settles, B., and McCallum, A. (2009). Active learning by labeling features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 81–90. Association for Computational Linguistics.
- Felt, P., Ringger, E., Seppi, K., Heal, K., Haertel, R., and Lonsdale, D. (2012). First results in a study evaluating pre-annotation and correction propagation for machineassisted syriac morphological analysis. In *Proceedings* of the International Conference on Language Resources and Evaluation (LREC), pages 878–885.
- Felt, P., Ringger, E. K., Seppi, K., Heal, K. S., Haertel, R. A., and Lonsdale, D. (In Press.). Evaluating machine-assisted annotation in under-resourced settings. *Language Resources and Evaluation*.
- Francis, N. and Kucera, H. (1979). Brown Corpus Manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US.
- Ganchev, K., Pereira, F., Mandel, M., Carroll, S., and White, P. (2007). Semi-automated named entity annotation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics: Linguistic Annotation Workshop*, pages 53–56.
- Ganchev, K., Graça, J. a., Gillenwater, J., and Taskar, B. (2010). Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049.
- Gippert, J., Himmelmann, N. P., and Mosel, U. (2006). *Essentials of Language Documentation*, volume 178. Walter de Gruyter.
- Grenoble, L. A. and Whaley, L. J. (1998). *Endangered Languages: Language Loss and Community Response*. Cambridge University Press.
- Hovy, E. and Lavid, J. (2010). Towards a 'science' of corpus annotation: a new methodological challenge for corpus linguistics. *International Journal of Translation*, 22(1):13–36.
- Kroch, A. S. (1989). Reflexes of grammar in patterns of language change. *Language Variation and Change*, 1(03):199–244.
- Liang, P., Jordan, M. I., and Klein, D. (2009). Learning from measurements in exponential families. In *Proceed*ings of the 26th Annual International Conference on Machine Learning, pages 641–648. ACM.
- Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.
- Marcus, M., Santorini, B., and Marcinkiewicz, M. (1995). Treebank-2: Ldc95t7.
- Nesselhauf, N. (2004). Learner corpora and their potential for language teaching. *How to Use Corpora in Language Teaching*, 12.
- Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359.

- Roth, D. and Yih, W.-t. (2004). A Linear Programming Formulation for Global Inference in Natural Language Tasks. Defense Technical Information Center.
- Sampson, G. (2008). The SUSANNE Analytic Scheme. http://www.grsampson.net/RSue.html. Accessed: 10/3/2012.
- Santorini, B. (1990). Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision). Technical report, Department of Computer & Information Science, University of Pennsylvania, Philadelphia, US.
- Settles, B. (2010). Active learning literature survey. University of Wisconsin, Madison.
- Sinclair, J. M. (2004). *How to use Corpora in Language Teaching*, volume 12. John Benjamins Publishing Company.
- Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pages 173–180. Association for Computational Linguistics.
- Wolpert, D. H. (1992). Stacked generalization. Neural networks, 5(2):241–259.