Learning How to Self-Learn: Enhancing Self-Training Using Neural Reinforcement Learning

Chenhua Chen, Yue Zhang
Westlake Institute for Advanced Study (WIAS)
No.18, Shilongshan Road, Cloud Town, Xihu District
Hangzhou, Zhejiang Province, China
{*chenchenhua,zhangyue*}*@wias.org.cn*

Yuze Gao
Singapore University of Technology and Design (SUTD)
8 Somapah Road
Singapore 487372
yuze.gao@outlook.com

***Abstract*—Self-training is a useful strategy for semi-supervised learning, leveraging raw texts for enhancing model performances. Traditional self-training methods depend on heuristics such as model confidence for instance selection, the manual adjustment of which can be expensive. In addition, characteristics of extra training sentences are not considered beyond the baseline method. To address these challenges, we propose a deep reinforcement learning method to learn self-training strategy automatically. Based on the neural representation of sentences and the hidden features from classifiers, a deep Q-network based model is designed to capture their linguistic characteristics and learn an optimal policy for instance selection automatically. Results show that our approach outperforms baseline self-training in terms of better performances and stability.**

***Keywords*-deep reinforcement learning; self-learning; named entity recognition; neural networks;**

I. Introduction

Self-training is a commonly used semi-supervised learning strategy that has been used for various natural language processing (NLP) tasks, such as named entity recognition (NER) [1], part-of-speech (POS) tagging [2], [3] and parsing [4], [5], [6], [7]. The basic idea is to augment the original training set with a set of automatic predictions. There have been different strategies for selecting automatically labeled data, the most typical one being the confidence values of the baseline models.

How to define and measure the confidence of predictions is crucial for a successful self-training approach. Traditional self-training solutions manually design task-specific heuristics [8], [9], [10], [11], [12]. This can lead to two drawbacks: 1) manual adjustment of instance selection strategy can be rather costly; 2) for the best effect on an unknown dataset, the source of information is limited to model confidence and a few other simple heuristics. However, linguistic characteristics of specific test sentences cannot be easily captured.

We aim to address these issues by leveraging neural models to represent test sentences, using reinforcement learning to automatically learn an instance selection strategy. In particular, by reordering sentences as text stream, a self-training approach can be regarded as a decision process, where each step decides whether the next incoming automatically labeled instances should be selected. Since no gold labels exist for instance selection, we use deep Q-network (DQN) [13], [14], [15] to learn the selection strategy automatically, on the basis of the performance improvements on a set of development data.

To better capture the linguistics of sentences in the state of the DQN network, we designed a CNN model to learn the neural representation of sentences and their marginal probability distribution. The hidden features from the taggers are also input to the DQN network to calculate the Q-value for sentence selection.

A major advantage of our method as compared to traditional self-learning is that instance-level characteristics can be combined with model-level confidence information using a neural model that is adaptively trained using reinforcement learning. Such information combination allows DQN to mitigate the bias between labeled and unlabeled datasets. As a result, the selection of automatically labeled instances is more helpful to improve the self-training performance.

On standard NER and POS tagging tasks, our method shows better performance compared to traditional self-training. In particular, the sentences selected by our solution provide more useful information to promote the self-learning performance. We release our code at http://github.com/CCSoleil/dqn_rlsl.

II. Related Work

Self Training is a simple semi-supervised algorithm that has shown its effectiveness in parsing [4], [5], [6], [7], part of speech tagging [2], [16], [3], named entity recognition [1], [17], sentiment classification [18], [19], [20], and other NLP tasks. The performance of the self-training algorithms strongly depends on how automatically labeled data is selected at each iteration of the training procedure. Most existing approaches set up a threshold and treat a set of unlabeled examples as the high-confident prediction if its prediction is above the pre-defined threshold value. Such a selection metric may not provide a reliable selection [21].

Some researchers explore extra metrics as an auxiliary measurement to evaluate instances from unlabeled data. For example, Ardehaly and Culotta [9] used coefficients learned from the model on the source domain as a selection metric and report a positive effect when applying self training in the target for hierarchical multi-label classification task. Katz-Brown et al. [10] proposed to produce a ranked list of n-best predicted parses and selected the one

Input: Train set T, dev set D, unlabeled data U, $budget$;
Output: $tagger$;
Initialize DQN; tagger $\leftarrow$ train(T, D);
$i \leftarrow 0$; $b \leftarrow 0$; $S \leftarrow \{\}$;
while $i < budget$ *and* U *is not empty* **do**
  while $b < batchsize$ **do**
    $x \leftarrow$ a random instance from U;
    $U \leftarrow U \backslash x$; $b \leftarrow b + 1$;
    $qvalue \leftarrow DQN(x, tagger)$;
    if $argmax(qvalue) == 1$ **then**
      $y \leftarrow tag(tagger, T)$;
      $T \leftarrow T \cup (x, y)$; $S \leftarrow S \cup (x, y)$;
    end
  end
  $tagger \leftarrow train(T, D)$;$R \leftarrow tagger(S)$;
  DQN $\leftarrow$ updateDQN$(tagger, R, S)$;
  $i \leftarrow i + 1$; $b \leftarrow 0$; $S \leftarrow \{\}$;
end

Algorithm 1: DQN-based self-training

yields the best external evaluation scored on the downstream external task (i.e., machine translation). Rosenberg et al. [8] examined a few selection metrics for object detection task, and showed that detector-independent metric outperforms the more intuitive confidence metric. Various pseudo-labeled example selection strategies [11], [12] have been proposed.

Zhou et al. [22] explore a guided search algorithm to find informative unlabeled data subsets in the self-training process. The experimental results demonstrate that the proposed algorithm is in general more reliable and more effective than the standard self-training algorithm. These heuristic choices however require careful parameter tuning and domain specific information.

Most recently, Levati et al. [23] proposed an algorithm to automatically identify an appropriate threshold from a candidate list for the reliability of predictions. The automatic selected threshold is used in the next iteration of self-training procedure. They scored each candidate threshold by evaluating whether the mean of the out of bag error between the examples with reliability score greater than the considered threshold is significantly different from the mean of the out of bag error of all the examples. Our work is in line, but extends the role that the neural network plays in guiding self-training.

Our work also belongs to a recent strand of work on **meta learning** [24], [25], [26], [27], [28], [29], [30], most of which leverage deep reinforcement learning. Fang et al. [15] leveraged a deep Q-network to learn the query policy for active learners. The substantial difference is that active learning needs to query human experts or oracles to provide gold labels, whereas self-training associates instances with automatic prediction labels. Wu et al. [31] investigated reinforced co-training for clickbait detection and text classification. Their approach learns how to choose predefined unlabeled subsets based on model-level confidence (i.e., the probability distributions of two base classifiers). Our work is different from theirs in three aspects. First, we adopt a stream-based learning strategy, randomly sampling every batch of instances. Second, we adopt a binary Q-value in DQN model instead of k values. Third, the state of DQN in our work is richer because our model uses not only the probability output of the baseline classifier, but also the representations of instances and internal neural features from the base classifier. Therefore, our model can potentially exploit more information to learn a stable strategy for instance selection. To our knowledge, we are the first to use meta learning for enhancing self-training.

III. Learning how to self-train

We design a deep reinforcement learning neural network to train the self-training function that can select high quality unlabeled instances automatically to improve the performance.

A. DQN-based self-training

Algorithm 1 shows pseudocode for DQN-based self-training, which consists of three main steps. First, it randomly accesses a batch of unlabeled instances, using the DQN model to assign confidence scores ($qvalue$) for each instance. Second, it makes decisions about instance acceptance or rejection. Once an instance is accepted, its prediction will be included to the training set T. Third, the tagger and the DQN are retrained using the updated training set and performance rewards. This process repeats until the predefined budget is used up or the algorithm traverses all the instances.

B. Model Structure

Fig. 1 shows the the three-layered neural network for Q-function ($qvalue$ in Algorithm 1) learning. The input layer is the state representation. In particular, a state s consists of four elements (h_s, h_c, h_p, h_t), where h_s is the content representation of the arriving instance, h_c is the tagging confidence of the instance, h_p is the marginals of the predictions for this instance, and h_t is the hidden features from the baseline neural tagger. h_s is the instance-level linguistic characteristics, h_c, h_p, h_t are model-level characteristics.

As shown in Fig. 1, each instance is represented by concatenated word embeddings. This representation vector is passed through a convolution neural network (CNN) with the convolution size being (3, 4, 5), and 128 filters for each size. Each filter uses a linear transformation with a rectified function. The filter outputs are then merged using a max-pooling operation to yield a hidden state h_s, resulting a vector of size 384 that represents the content of this instance.

The confidence of the tagger for an instance h_c is defined based on the most probable label sequence for this instance. We adopt a Bi-LSTM-CRF tagger [32] as the baseline tagger and the confidence value can be calculated via forward-backward algorithm. This value also can be defined for different taggers.

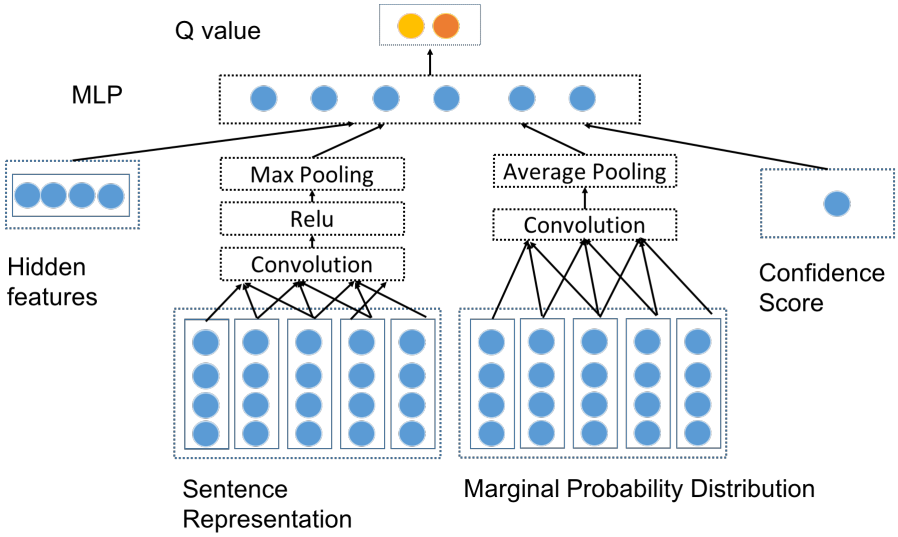


Figure 1. The neural network for Q-function.

The marginals of the prediction for an instance is also passed to a CNN, with the filter number being 20 and the convolution size being 3. These feature maps are then sub-sampled with mean pooling to capture the average uncertainty in each filter. The pooling result h_p is used to represent the predictive marginals. Finally, $s = (h_s, h_c, h_p, h_t)$ is fed to the second layer with a rectified function, the dimension of which is 256. The output feature is then used to calculate the expected Q-value in a softmax layer of size 2, indicating whether an instance should be selected or not.

In order to include richer model-level characteristics, we extract the last hidden layers from the tagger to represent the view of the tagger for the arriving instance.

C. Training DQN

DQN learns an optimal policy π via a Q-function: $Q^\pi(s,a) \rightarrow R$, where s is the current state, a is the action, and R is the cumulative reward after taking a from s. $Q^\pi(s,a)$ can be iteratively updated using the rewards obtained from the sequence of actions, based on the Bellman equation. Formally, $Q^\pi(s,a) = \mathbb{E}^\pi[\sum_{t=i}^{T} \gamma^{t-i} r_t | s_i = s, a_i = a]$, where $\gamma \in [0,1]$ is the discounting rate and r_t is the reward at the t-th step.

Following Minh et al. [13], we train DQN using an experience replay mechanism. The current state, its actions and corresponding rewards are recorded in a memory. The parameters of the DQN are learned using SGD to match the Q-values predicted by the DQN and the expected Q-values from the Bellman equation, $r_i + \gamma \max_a Q(s_{i+1}, a; \theta)$. Samples are randomly selected from the experience memory to update the parameters of DQN by minimizing the loss function:

$$\mathbb{L}(\theta) = (r + \gamma \max_{a'} Q(s', a') - Q(s, a))^2$$

Here s' is the new state after executing a from s. The training procedure is repeated with incoming instances. We conduct a significance test on the performance difference among a series of consecutive actions to decide whether an episode should be terminated. If the performance difference of the tagger is insignificant, we restart a new episode.

Rewards. If an instance is not selected, the reward is set to 0; otherwise, the reward of a select action is defined as the performance difference of the tagger after a batch of instances are added to the training set.

IV. Experiments

We conduct a series of NER and POS tasks to evaluate our proposal.

A. Data

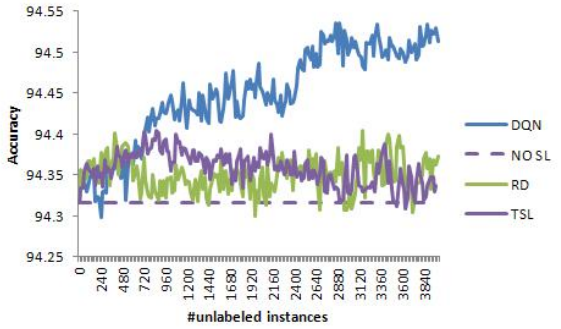
For the NER experiments, we use the training, development and evaluation data sets from the CoNLL 2002/2003 shared tasks [33], [34] for four languages: English, German, Spanish and Dutch. We follow existing corpus partitions, with `train` used for policy training, `testb` used as development set for computing rewards, and final results are reported on `testa`.

The Europarl-v7 raw dataset [35] for machine translation is selected as the unlabeled data for the aforementioned four languages. A pretrained embedding for each language [36] is used in the experiments.

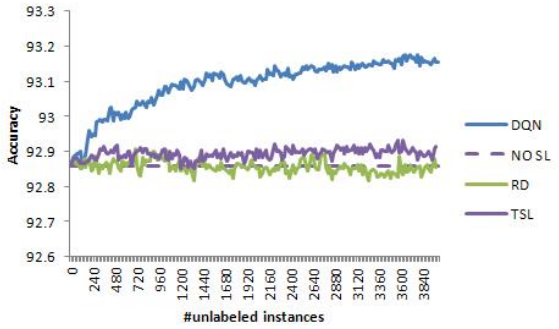
For the POS experiments, we use the SANCL2012 dataset [37]. In this dataset, the OntoNotes are used as the labeled data, and gweb-newsgroups, gweb-reviews, gweb-emails, gweb-weblogs are selected as unlabeled data.

B. Baselines and Training Settings

We implement two baseline solutions, i.e., a random sampling solution (denoted as `RD`) and a confidence-based self-training solution (denoted as `TSL`). For both `RD` and `TSL`, the tagger is initialized with the same instances as the DQN-based approach. For `RD`, a stream of unlabeled sentences are randomly selected and predicted by the



(a) Newsgroups POS Tagging (training)



(b) Emails POS Tagging (training)

Figure 2. Self-training for POS tagging (training)

	Newsgroup	Reviews	Weblogs	Emails	Average
NO SL	94.69	92.40	94.94	93.20	93.81
RD	94.81	92.63	94.94	93.40	93.94
TSL	94.83	92.65	95.00	93.41	93.97
DQN	**94.87**	**92.71**	**95.04**	**93.47**	**94.02**

Table I
POS TAGGING PERFORMANCES

	English	Dutch	Spanish	German	Average
NO SL	92.44	82.54	84.52	78.00	84.38
RD	92.43	82.62	84.40	77.93	84.35
TSL	92.64	83.01	84.65	78.35	84.66
DQN	**92.74**	**83.57**	**84.80**	**78.68**	**84.95**

Table II
NER PERFORMANCES

tagger. For TSL, the confidence score of an unlabeled sentence x_i with n tokens is $\sqrt[n]{\max_y P(y|x_i)}$.

For the DQN solution, an episode of DQN terminates when the latest 10 rewards is smaller than a given threshold (0.001), indicating insignificant performance change. To optimize the weights of the DQN, the DQN for each task is trained for 10,000 episodes. In the experiments, the batch size is set to 32, γ is set to 0.99, the replay buffer size is set to 1,000 and ϵ-greedy sampling is used.

In the NER experiments, we use the state-of-the-art BiLSTM-CNN-CRF NER tagger[1] [32] as our baseline model. In the POS experiments, a CRF model is used to implement the POS tagger, since it gives competitive results and run faster.

C. Self-training for NER

Table II shows the results. In general, RD does not outperform NO SL. TSL is better than both NO SL and RD, in particular improving an F1-score of 0.3 compared with NO SL. DQN behaves consistently with the best performances on all the four datasets. Compared with NO SL, DQN has an 0.57 improvement of F1-score on average. In particular, the improvement of F1-score over NO SL is more than 1.0 for Dutch. Compared with TSL, DQN is significantly better on all four language datasets. In addition, we find that DQN is more stable than RD and TSL. After adding more unlabeled sentences, the F1-scores of the two baselines drop dramatically, whereas DQN still has a stable performance.

D. Self-learning for POS tagging

The results are shown in Table I. We can observe that our DQN-based solution has a better performance than the two baseline solutions in each domain. In particular, for the Reviews scenario, the accuracy of the our DQN-solution increases about 0.31% compared to $NOSL$ solution. The accuracy of the two baselines also increase with 0.23% and 0.25% respectively, and the accuracy of the DQN-solution is further better than the TSL and RD solutions. Similar results can be observed for the Newsgroup, Weblogs and Emails scenarios, as shown in Table I.

[1]https://github.com/guillaumegenthial/sequence_tagging

E. Training characteristics

We also recorded the performance of the tagger during the training. Fig. 2 shows the training performance of the pos-tagging scenarios. We can observe that the performance of the DQN-solution increases slowly by adding the increasing number of unlabeled instances in all four scenarios (up to 0.3%). This shows that the DQN-based self-learning is effective in improving the training performance. Similar training trends can be observed for the NER. This is because our DQN-solution converges slowly than traditional method. As a result, our DQN-solution is able to utilize more unlabeled sentences effectively to promote the self-training performance.

V. ANALYSIS

We investigate the sentences selected by each solution, and find that TSL selects the sentences most with no entities (around 95.6% of the selected sentences) for English. This might be due to the fact that sentences with no entities are usually associated with larger confidence values. In contrast, DQN favors the sentences with richer entities, resulting that around 54.1% of the selected sentences contain at least one entity. Similar results are observed for Spanish and German.

ID	Instance	TSL	DQN
1	[Wayne Ferrei ra]$_{PER}$ [South Africa]$_{LOC}$ beat [Jiri Novak]$_{PER}$ [Czech]$_{LOC}$.	Yes	Yes
2	The rapporteur wants assistants working in [Brussels]$_{LOC}$ to be covered by [Community rules]$_{ORG}$.	No	Yes
3	says is first state to apply for new welfare.	Yes	No
4	Commissioner [Frattini]$_{PER}$ wants [Europe]$_{LOC}$ to attract a skilled workforce.	No	Yes

Table III
SELECTED INSTANCES OF ENGLISH NER TASK (BLUE: CORRECT PREDICTION, VIOLET: WRONG ANSWER.)

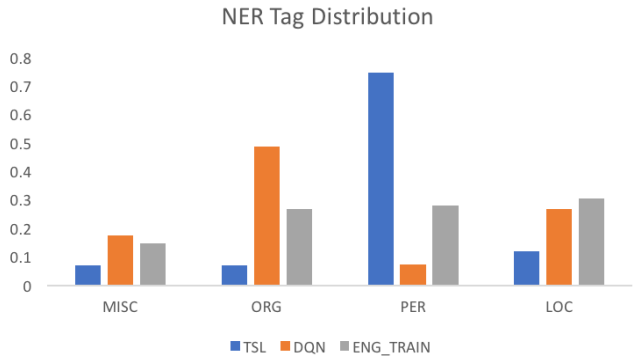


Figure 3. NER tag distributions on instances selected by TSL, instances selected by DQN and the original English training dataset.

Table III shows some instances selected by TSL and DQN for the English NER task. All entities in Instance 1 are tagged accurately. Both TSL and DQN select this instance. *Community rules* is wrongly tagged as ORG in Instance 2, leading to a small confidence score. As a result, TSL did not select this useful instance. In contrast, DQN successfully accepts this prediction result. The LOC entity in this instance is correctly tagged. A similar phenomenon can be found in Instance 4, both the PER and LOC entities are tagged with a very small confidence score, TSL rejects this instance, whereas DQN accepts this prediction. TSL assigns Instance 3 a very large confidence score since there are no named entities at all, while DQN learns to reject this instance, which is not useful for self training.

In addition, compared with TSL, DQN produces a much closer entity tag distribution to that of the original English dataset, as shown in Fig 3. TSL produces surprisingly large number of PER entities and small number of MISC and ORG entities, while DQN gives less spiky distributions.

VI. CONCLUSION

We proposed a deep Q-network for automatically learning to self learn, accepting model predictions based on instance-level linguistic characteristics and model-level outputs. Results on NER and POS tagging tasks show that our approach is consistently better than the random sampling baseline and the traditional confidence-based self-learning solutions.

REFERENCES

[1] Z. Kozareva, B. Bonev, and A. Montoyo, "Self-training and co-training applied to spanish named entity recognition," in *Mexican International Conference on Artificial Intelligence*. Springer, 2005, pp. 770–779.

[2] W. Wang, Z. Huang, and M. Harper, "Semi-supervised learning for part-of-speech tagging of mandarin transcribed speech," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–137.

[3] Y. Qi, P. Kuksa, R. Collobert, K. Sadamasa, K. Kavukcuoglu, and J. Weston, "Semi-supervised sequence labeling with self-learned features," in *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE, 2009, pp. 428–437.

[4] D. McClosky, E. Charniak, and M. Johnson, "Effective self-training for parsing," in *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, ser. HLT-NAACL '06. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006, pp. 152–159.

[5] D. McClosky, E. Charniak, and M. Johnson, "When is self-training effective for parsing?" in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2008, pp. 561–568.

[6] Z. Huang and M. Harper, "Self-training pcfg grammars with latent annotations across languages," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, ser. EMNLP '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 832–841.

[7] K. Sagae, "Self-training without reranking for parser domain adaptation and its impact on semantic role labeling," in *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, ser. DANLP 2010. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 37–44.

[8] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," in *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, vol. 1. IEEE Press, Jan 2005, pp. 29–36.

[9] E. M. Ardehaly and A. Culotta, "Domain adaptation for learning from label proportions using self-training," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, pp. 3670–3676.

[10] J. Katz-Brown, S. Petrov, R. McDonald, F. Och, D. Talbot, H. Ichikawa, M. Seno, and H. Kazawa, "Training a parser for machine translation reordering," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, July 2011, pp. 183–192.

[11] B. Medlock and T. Briscoe, "Weakly supervised learning for hedge classification in scientific literature," in *ACL*, vol. 2007, 2007, pp. 992–999.

[12] H. Daumé III, "Cross-task knowledge-constrained self training," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2008, pp. 680–688.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[14] H. Guo, "Generating text with deep reinforcement learning," *arXiv preprint arXiv:1510.09202*, 2015.

[15] M. Fang, Y. Li, and T. Cohn, "Learning how to active learn: A deep reinforcement learning approach," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, September 2017, pp. 595–605.

[16] Z. Huang, V. Eidelman, and M. Harper, "Improving a simple bigram hmm part-of-speech tagger by latent annotation and self-training," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. Boulder, Colorado: Association for Computational Linguistics, June 2009, pp. 213–216.

[17] Q. Liu, B. Liu, D. Wu, Y. Liu, and X. Cheng, "A self-learning template approach for recognizing named entities from web text," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Nagoya, Japan: Asian Federation of Natural Language Processing, October 2013, pp. 1139–1143.

[18] V. Van Asch and W. Daelemans, "Predicting the effectiveness of self-training: Application to sentiment classification," *arXiv preprint arXiv:1601.03288*, 2016.

[19] B. Drury, L. Torgo, and J. J. Almeida, "Guided self training for sentiment classification," in *Proceedings of Workshop on Robust Unsupervised and Semisupervised Methods in Natural Language Processing*, Hissar, Bulgaria, September 2011, pp. 9–16.

[20] Z. Liu, X. Dong, Y. Guan, and J. Yang, "Reserved self-training: A semi-supervised sentiment classification method for chinese microblogs," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Nagoya, Japan: Asian Federation of Natural Language Processing, October 2013, pp. 455–462.

[21] M. Chen, K. Q. Weinberger, and J. Blitzer, "Co-training for domain adaptation," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2456–2464.

[22] Y. Zhou, M. Kantarcioglu, and B. Thuraisingham, "Self-training with selection-by-rejection," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 795–803.

[23] J. Levati, M. Ceci, D. Kocev, and S. Deroski, "Self-training for multi-target regression with tree ensembles," *Know.-Based Syst.*, vol. 123, no. C, pp. 41–60, May 2017.

[24] M. Andrychowicz, M. Denil, S. G. Colmenarejo, M. W. Hoffman, D. Pfau, T. Schaul, and N. de Freitas, "Learning to learn by gradient descent by gradient descent," *CoRR*, vol. abs/1606.04474, 2016. [Online]. Available: http://arxiv.org/abs/1606.04474

[25] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, "Rl$ˆ2$: Fast reinforcement learning via slow reinforcement learning," *CoRR*, vol. abs/1611.02779, 2016. [Online]. Available: http://arxiv.org/abs/1611.02779

[26] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick, "Learning to reinforcement learn," *CoRR*, vol. abs/1611.05763, 2016. [Online]. Available: http://arxiv.org/abs/1611.05763

[27] P. Bachman, A. Sordoni, and A. Trischler, "Learning algorithms for active learning," *CoRR*, vol. abs/1708.00088, 2017. [Online]. Available: http://arxiv.org/abs/1708.00088

[28] S. Yeung, V. Ramanathan, O. Russakovsky, L. Shen, G. Mori, and F. Li, "Learning to learn from noisy web videos," *CoRR*, vol. abs/1706.02884, 2017. [Online]. Available: http://arxiv.org/abs/1706.02884

[29] M. Woodward and C. Finn, "Active one-shot learning," *CoRR*, vol. abs/1702.06559, 2017. [Online]. Available: http://arxiv.org/abs/1702.06559

[30] Y. Wei, Y. Zhang, J. Huang, and Q. Yang, "Transfer learning via learning to transfer," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmssan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 5072–5081. [Online]. Available: http://proceedings.mlr.press/v80/wei18a.html

[31] J. Wu, L. Li, and W. Y. Wang, "Reinforced co-training," *arXiv preprint arXiv:1804.06035*, 2018.

[32] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *north american chapter of the association for computational linguistics*, pp. 260–270, 2016.

[33] E. F. Tjong Kim Sang, "Introduction to the conll-2002 shared task: Language-independent named entity recognition," *arXiv preprint cs/0209010*, 2002.

[34] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003, pp. 142–147.

[35] P. Koehn, "Europarl: A parallel corpus for statistical machine translation," in *MT summit*, vol. 5, 2005, pp. 79–86.

[36] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[37] S. Petrov and R. McDonald, "Overview of the 2012 shared task on parsing the web." Citeseer, 2012.