Active^2 Learning: Actively reducing redundancies in Active Learning methods for Sequence Tagging

Rishi Hazra
rishihazra@iisc.ac.in
Indian Institute of Science
Bangalore, 560012, INDIA

Shubham Gupta
shubhamg@iisc.ac.in
Indian Institute of Science
Bangalore, 560012, INDIA

Ambedkar Dukkipati
ambedkar@iisc.ac.in
Indian Institute of Science
Bangalore, 560012, INDIA

Abstract

While deep learning is a powerful tool for natural language processing (NLP) problems, successful solutions to these problems rely heavily on large amounts of annotated samples. However, manually annotating data is expensive and time-consuming. Active Learning (AL) strategies reduce the need for huge volumes of labeled data by iteratively selecting a small number of examples for manual annotation based on their estimated utility in training the given model. In this paper, we argue that since AL strategies choose examples independently, they may potentially select similar examples, all of which may not contribute significantly to the learning process. Our proposed approach, Active$^\mathbf{2}$ Learning (A$^\mathbf{2}$L), actively adapts to the deep learning model being trained to further eliminate such redundant examples chosen by an AL strategy. We show that A$^\mathbf{2}$L is widely applicable by using it in conjunction with several different AL strategies and NLP tasks. We empirically demonstrate that the proposed approach is further able to reduce the data requirements of state-of-the-art AL strategies by an absolute percentage reduction of $\approx \mathbf{3-25}\%$ on multiple NLP tasks while achieving the same performance [1].

1 Introduction

Active Learning (AL) (Freund et al., 1997; McCallum and Nigam, 1998) reduces the need for large quantities of labeled data by intelligently selecting unlabeled examples for expert annotation in an iterative process. Many Natural Language Processing (NLP) tasks like named entity recognition, part-of-speech tagging, etc., are very data-intensive and require a meticulous, time-consuming, and costly annotation process. On the other hand, unlabeled data is practically unlimited. Due to this, many researchers have explored applications of active learning for NLP (Thompson et al., 1999; Figueroa et al., 2012). A general AL method proceeds as follows: **(i)** The partially trained model for a given task is used to (possibly incorrectly) annotate the unlabeled examples. **(ii)** An *active learning strategy* selects a subset of the newly labeled examples via a criterion that quantifies the perceived utility of examples in training the model. **(iii)** The experts verify/improve the annotations for the selected examples. **(iv)** These examples are added to the training set and the process repeats. AL strategies differ in the criterion used in step **(ii)** (see Section 4.1).

We claim that *all* AL strategies select redundant examples in step **(ii)**. If one example satisfies the selection criterion, then many other *similar* examples will also satisfy it (see the next paragraph for details). As the examples are selected independently, AL strategies redundantly choose all of these examples even though, in practice, it is enough to label only a few of them (ideally just one) for training the model. This leads to higher annotation costs, wastage of resources, and reduces the effectiveness of AL strategies. In this paper, we address this problem by proposing a new approach called A$^\mathbf{2}$L (read as active-squared learning). Based on A$^\mathbf{2}$L, we propose two methods and empirically establish that they further reduce the data requirements of state-of-the-art AL strategies.

Any approach for eliminating redundant examples must have the following qualities: **(i)** The redundancy should be evaluated in the context of the model being trained. **(ii)** The approach should apply to a wide variety of commonly used models in NLP. **(iii)** It should be compatible with several existing AL strategies. The first point merits more explanation. As a model is trained, depending on the downstream task, it learns to focus on certain properties of the input. Examples that share these properties (for instance, the sentence structure) are similar from the perspective of the model. If the

[1] accepted to Findings of EMNLP 2020

model is confused about one such example, it will likely be confused about all of them. However, it is often enough to label only a few of these examples to train the model. We refer to a similarity measure that computes similarity in the context of a model as a *model-aware similarity measure* (Section 3.1).

Contributions: (i) We propose a Siamese twin's (Bromley et al., 1994; Mueller and Thyagarajan, 2016) based method for computing model-aware similarity to eliminate redundant examples chosen by an AL strategy. This Siamese network actively adapts itself to the underlying model as the training progresses. We then use clustering based on similarity scores to eliminate redundant examples. **(ii)** We develop a second, computationally more efficient approach that approximates the first one with a minimal drop in performance by avoiding the clustering step. Both of these approaches have the desirable properties mentioned above. **(iii)** We experiment with several AL strategies and NLP tasks to empirically demonstrate that our approaches are widely applicable and significantly reduce the data requirements of state-of-the-art AL strategies while achieving the same performance. To the best of our knowledge, we are the first to identify the importance of model-aware similarity, and exploit it to address the problem of redundancy in AL.

2 Related Work

Active learning has a long and successful history in the field of machine learning (Dasgupta et al., 2009; Awasthi et al., 2017). However, as the learning models have become more complex, especially with the advent of deep learning, the known theoretical results for active learning are no longer applicable (Shen et al., 2018). This has prompted the development of a diverse range of heuristics to adapt the active learning framework to deep learning models (Shen et al., 2018). Many AL strategies have been proposed (Sha and Saul, 2007; Blundell et al., 2015; Gal and Ghahramani, 2016a), however, since they choose the examples independently, the problem of redundancy (Section 1) applies to all of them. The proposed approach is compatible with any AL strategy and we experiment with some common AL strategies in Section 4.

We experiment with a variety of NLP tasks like named entity recognition (Nadeau and Sekine, 2007), part-of-speech tagging (Marcus et al., 1993), and so on (Tjong Kim Sang and Buchholz, 2000; Landes and Leacock, 1998). The tasks chosen by us form the backbone of many practical information extraction problems. Many deep learning models have recently advanced the state-of-art for these tasks. Our proposed approach is compatible with any NLP model, provided it supports the usage of an AL strategy and hence, we experiment with two sequence tagging models borrowed from (Siddhant and Lipton, 2018) and (Lample et al., 2016). In the chosen models, we use Conditional Random Fields (CRF) (Lample et al., 2016) to compute measures of uncertainty needed by the AL strategies (Section 4.1). Many recent attempts at applying active learning to sequence tagging have been made (Siddhant and Lipton, 2018), however, the issue of redundancy (Section 1) has largely been ignored.

Existing approaches have used model independent similarity scores to promote diversity in the chosen examples. For instance, in Chen et al. (2015), the authors use cosine similarity to precalculate pairwise similarity between examples. We instead argue in the favor of model-aware similarity scores and learn an expressive notion of similarity using neural networks. We compare our approach with a modified version of this baseline using cosine similarity on Infersent embeddings (Conneau et al., 2017).

3 Proposed Approaches

We use $\mathcal{M}$ to denote the model being trained for a given NLP task and assume that $\mathcal{M}$ has a module called encoder for encoding the input sentences. For example, for a named entity recognition task, the encoder in $\mathcal{M}$ may be modeled by an LSTM network (Hochreiter and Schmidhuber, 1997). We do not require the encoder to have a specific architecture as the network architecture used in our approaches can be modified accordingly independent our conceptual contribution.

3.1 Model-Aware Similarity Computation

A measure of similarity between examples is required to discover redundancy. The simplest solution is to compute the cosine similarity between input sentences (Chen et al., 2015; Shen et al., 2018) using, for instance, the InferSent encodings (Conneau et al., 2017). However, sentences that have a low cosine similarity may still be similar in the context of the downstream task. Model $\mathcal{M}$ has no incentive to distinguish among such examples. A good strategy is to label a diverse set of sentences from the perspective of the model. For example, it

```
Algorithm 1: Active$^2$ Learning
  Data: $\mathcal{D}_1$: task dataset;
        $\mathcal{D}_2$: auxiliary similarity dataset
  Input: $\bar{\mathcal{D}} \leftarrow (D_1, \ldots, D_l)$: Partitioning of
         unlabeled data, each $D_i$ is a set.
  Output: Labeled data
Initialization: $\mathcal{D} \longleftarrow 2\%$ of dataset $\mathcal{D}_1$;
$\mathcal{D} \longleftarrow$ ANNOTATE($\mathcal{D}$);
$\mathcal{M} \longleftarrow$ TRAIN($\mathcal{D}$);
$\mathcal{M}_{A^2L} \longleftarrow$ TRAIN($\mathcal{M}(\mathcal{D}_2)$);
for $i \leftarrow 1$ to $l$ do
    $\mathcal{S} \leftarrow \mathcal{AL}(\mathrm{D}_i)$;    // confused samples
    if // Model-Aware Siamese
    then
        for each pair $(\mathrm{s}_m, \mathrm{s}_n)$ in $\mathcal{S}$ do
            $\mathbf{S}[m, n] \leftarrow \mathcal{M}_{A^2L}(\mathrm{s}_m, \mathrm{s}_n)$;
        $\mathcal{R} \longleftarrow$ CLUSTER($\mathbf{S}$);
    else
        // Integrated Clustering
        $\mathcal{R} \longleftarrow \mathcal{M}_{A^2L}(\mathcal{S})$;
    $\mathcal{R} \longleftarrow$ ANNOTATE($\mathcal{R}$);
    $\mathcal{D} \longleftarrow \mathcal{D} \cup \mathcal{R}$;
    $\mathcal{M} \longleftarrow$ RETRAIN($\mathcal{D}$)
```

is unnecessary to label sentences that use different verb forms but are otherwise similar, if the task is agnostic to the tense of the sentence. Hence, it is important to use model-aware similarity to reduce redundancy. A straightforward extension of cosine similarity to the encodings generated by model $\mathcal{M}$ achieves this. However, a simplistic approach like this would likely be incapable of discovering complex similarity patterns in the data. Next, we describe two approaches that use more expressive model-aware similarity measures. In Section 4, we compare them with the baselines mentioned above on several NLP tasks to validate our claims.

3.2 Model-Aware Siamese

In this approach, we use a Siamese twin's network (Bromley et al., 1994) to compute the pairwise similarity between encodings obtained from model $\mathcal{M}$. A Siamese twin's network consists of an encoder (called the Siamese encoder) that feeds on the output of model $\mathcal{M}$'s encoder. The outputs of Siamese encoder are used for computing the similarity between each pair of examples a and b as:

$$\mathrm{sim}(a, b) = \exp\left(-||\mathbf{o}^a - \mathbf{o}^b||_2\right), \qquad (1)$$

where $\mathbf{o}^a$ and $\mathbf{o}^b$ are the outputs of the Siamese encoder for sentences a and b respectively. Let N denote the number of examples chosen by an AL strategy. We use the Siamese network to compute the entries of an $N \times N$ similarity matrix $\mathbf{S}$ where the entry $S_{ab} = \mathrm{sim}(a, b)$. We then use the spectral clustering algorithm (Ng et al., 2002) on the similarity matrix $\mathbf{S}$ to group similar examples. A fixed number of examples from each cluster are added to the training dataset after annotation by experts.

We train the Siamese encoder to predict the similarity between sentences from SICK (Sentences Involving Compositional Knowledge) dataset (Marelli et al., 2014) using mean squared error. This dataset contains pairs of sentences with manually annotated similarity scores. The sentences are encoded using the encoder in $\mathcal{M}$ and then passed on to the Siamese encoder for computing similarities. The encoder in $\mathcal{M}$ is kept fixed while training the Siamese encoder. The trained Siamese encoder is then used for computing similarity between sentences selected by an AL strategy for the given NLP task as described above. As $\mathcal{M}$ is trained over time, the distribution of its encoder output changes and hence we periodically retrain the Siamese network to sustain its model-awareness.

The architecture of Siamese encoder is dependent on the architecture of model $\mathcal{M}$'s encoder. The number of clusters and the number of examples drawn from each cluster are user-specified hyper-parameters. The similarity computation can be done efficiently by computing the output of Siamese encoder for all N examples before evaluating equation 1, instead of running the Siamese encoder $O(N^2)$ times. The clustering algorithm runs in $O(N^3)$ time. For an AL strategy to be useful, it should select a small number of examples to benefit from interactive and intelligent labeling. We expect N to be small for most practical problems, in which case the computational complexity added by our approach would only be a small fraction of the overall computational complexity of training the model with active learning.

3.3 Integrated Clustering Model

While the approach described in Section 3.2 works well for small to moderate values of N, it suffers from a computational bottleneck when N is large. We integrate the clustering step into the similarity computation step to remedy this and call the resultant approach as Integrated Clustering Model (*Int*

Model). Here, the output of model $\mathcal{M}$'s encoder is fed to a clustering neural network $\mathcal{C}$ that has K output units with the softmax activation function. These units correspond to the K clusters and each example is directly assigned to one of the clusters based on the softmax output.

To train the network $\mathcal{C}$, we choose a pair of similar examples (say a and b) and randomly select a negative example (say c) from the Quora Pairs dataset[3]. This dataset contains information about duplicate questions on Quora. All examples are encoded via the encoder of model $\mathcal{M}$ and then passed to network $\mathcal{C}$. The unit with the highest probability value for example a is treated as the ground-truth class for example b. The objective is to maximize the probability of example b belonging to its ground truth class while minimizing the probability of example c belonging to the same class:

$$\begin{aligned}\mathcal{L}(a,b,c) = &- \lambda_1 \log p^b_{i_a} - \lambda_2 \log(1 - p^c_{i_a}) \\ &+ \lambda_3 \sum_{k=1}^{K} p^b_k \log p^b_k. \end{aligned} \tag{2}$$

Here λ_1, λ_2, and λ_3 are user-specified hyperparameters, p^x_j is the softmax output of the j^{th} unit for example x, $j = 1, 2, \ldots, K$, $x = a, b, c$, and $i_a = \arg\max_{j \in \{1,2,\ldots K\}} p^a_j$. The third term encourages the utilization of all the K units across examples in the dataset. As before, a trained network $\mathcal{C}$ is used for clustering examples chosen by an AL strategy, and we select a fixed number of examples from each cluster for manual annotation.

It is important to note that: **(i)** These methods are not AL strategies. Rather, they can be used in conjunction with any existing AL strategy. Moreover, given a suitable Siamese encoder or clustering network $\mathcal{C}$, they apply to any model $\mathcal{M}$. **(ii)** Our methods compute model-aware similarity since the input to the Siamese or the clustering network is encoded using the model $\mathcal{M}$. The proposed networks also adapt to the underlying model as the training progresses. Algorithm 1 describes our general approach called Active^2 Learning.

4 Experiments

We establish the effectiveness of our approaches by demonstrating that they: **(i)** are compatible with three of the most popular AL strategies, **(ii)** work well across a variety of NLP tasks and models, and **(iii)** further reduce the data requirements of existing AL strategies, while achieving the same performance. We begin by describing the AL strategies used in our experiments.

4.1 Active Learning Strategies

Margin based AL strategy: Let $s(\mathbf{y}) = \mathrm{P}_{\boldsymbol{\theta}}(\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x})$ be the score assigned by a model $\mathcal{M}$ with parameters $\boldsymbol{\theta}$ to output $\mathbf{y}$ for a given example $\mathbf{x}$. Margin is defined as the difference in scores obtained by the best scoring output $\mathbf{y}$ and the second best scoring output $\mathbf{y}'$, i.e.:

$$\mathrm{M}_{\text{margin}} = \max_{\mathbf{y}} s(\mathbf{y}) - \max_{\mathbf{y}' \neq \mathbf{y}_{\max}} s(\mathbf{y}'), \tag{3}$$

where, $\mathbf{y}_{\max} = \arg\max_{\mathbf{y}} s(\mathbf{y})$. We empirically determine a hyper-parameter threshold τ_1 and select all examples for which $\mathrm{M}_{\text{margin}} \leq \tau_1$. We use Viterbi's algorithm (Ryan and Nudd, 1993) to compute the scores $s(\mathbf{y})$ in our experiments.

Entropy-based AL Strategy: All the NLP tasks that we consider require the model $\mathcal{M}$ to produce an output for each token in the sentence. Let $\mathbf{x}$ be an input sentence that contains $n(\mathbf{x})$ tokens and define $\bar{s}_j = \max_{o \in \mathcal{O}} \mathrm{P}_{\boldsymbol{\theta}}(y_j = o|\mathbf{X} = \mathbf{x})$ to be the probability of the most likely output for the j^{th} token in $\mathbf{x}$. Here $\mathcal{O}$ is set of all possible outputs and y_j is the output corresponding to the j^{th} token in $\mathbf{x}$. We define the *normalized entropy score* as:

$$\mathrm{M}_{\text{entropy}} = -\frac{1}{n(\mathbf{x})} \sum_{j=1}^{n(\mathbf{x})} \bar{s}_j(\mathbf{y}) \log \bar{s}_j(\mathbf{y}). \tag{4}$$

Empirically, it seems important to normalize the entropy by the example length $n(\mathbf{x})$ as $\mathrm{M}_{\text{entropy}}$ is correlated with $n(\mathbf{x})$, and it may be undesirable to annotate longer length examples (Claveau and Kijak, 2017). The strategy selects examples with $\mathrm{M}_{\text{entropy}} \geq \tau_2$, where τ_2 is a hyper-parameter.

Bayesian Active Learning by Disagreement (BALD): Due to stochasticity, models that use dropout (Srivastava et al., 2014) produce a different output each time they are executed. BALD (Houlsby et al., 2011) exploits this variability in the predicted output to compute model uncertainty. Let $\mathbf{y}^{(t)}$ denote the best scoring output for $\mathbf{x}$ in the t^{th} forward pass, and let N be the number of forward passes with a fixed dropout rate, then:

$$\mathrm{M}_{\text{bald}} = 1 - \frac{\text{count}(\text{mode}(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)}))}{N}. \tag{5}$$

Task	Dataset	#Train/#Test	Example (Input/Output)
NER	CoNLL 2003	14987 / 3584	Fischler proposed EU measures after reports from Britain B-PER 0 B-MISC 0 0 0 0 B-LOC
POS	CoNLL 2003	14987 / 3584	He ended the World Cup on the wrong note PRP VBD DT NNP NNP IN DT JJ NN
CHUNK	CoNLL 2000	8936 / 2012	The dollar posted gains in quiet trading B-NP I-NP B-VP B-NP B-PP B-NP I-NP
SEMTR	SEMCOR[2]	13851 / 4696	This section prevents the military departments 0 Mental Agentive 0 0 Object
AUX	SICK	9000/1000	(1) Two dogs are fighting. (2) Two dogs are wrestling and hugging. Similarity Score: 4 (out of 5)
AUX	Quora Pairs[3]	16000 / 1000 (sets) [4]	(1) How do I make friends? (2) How to make friends? Label: 1

Table 1: Task and dataset descriptions. AUX is the task of training the Siamese network (Section 3.2) or Integrated network $\mathcal{C}$ (Section 3.3). **Citations**: CoNLL 2003 (Sang and De Meulder, 2003), CoNLL 2000 (Tjong Kim Sang and Buchholz, 2000), SEMCOR[2], SICK (Marelli et al., 2014), Quora Pairs[3].

Here the $\text{mode}(.)$ operation finds the output which is repeated most often among $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)}$, and the $\text{count}(.)$ operation counts the number of times this output was encountered. This strategy selects examples with $\text{M}_{\text{bald}} \geq \tau_3$, where τ_3 is a hyper-parameter.

4.2 Natural Language Processing Tasks

We experiment with several NLP tasks to demonstrate the wide applicability of our approach. Table 1 lists these tasks and information about the corresponding datasets used in our experiments. Table 1 also lists the two auxiliary datasets that we use for training the Siamese network (Section 3.2) and clustering network (Section 3.3). These tasks were primarily chosen because they require the experts to annotate each token in the sentence, which is especially challenging. It is easy to adapt our approach to other NLP tasks by suitably modifying the network architecture presented next. The Siamese/Clustering networks need continuous/binary similarity measures respectively, hence we have used two auxiliary datasets.

4.3 Details about Training

We use two sequence tagging architectures: CNN-BiLSTM-CRF model (CNN for character-level encoding and BiLSTM for word-level encoding) and a BiLSTM-BiLSTM-CRF model (BiLSTM for both character-level and word-level encoding) for all tasks Lample et al. (2016); Siddhant and Lipton (2018). These models were chosen for their performance and ease of implementation. The only architectural detail about these models that concerns us is that they use an LSTM which emits an output encoding for each token in the input sentence. See Appendix C for more details.

The Siamese network used for model-aware similarity computation (Section 3.2) consists of two bidirectional LSTM (BiLSTM) encoders. We pass each sentence in the pair from the SICK dataset to model $\mathcal{M}$ and feed the resulting encodings to a randomly chosen Siamese BiLSTM encoder. The output is a concatenation of terminal hidden states of the forward and backward LSTMs, which is used to compute the similarity score using (1). As noted before, we keep model $\mathcal{M}$ fixed while training the Siamese encoders, and use the trained Siamese encoders for computing similarity between examples chosen by an AL strategy. We maintain the model-awareness of the Siamese network by retraining it after every 10 iterations.

The architecture of the clustering model $\mathcal{C}$ (Section 3.3) is similar to that of the Siamese encoder. Additionally, it has a linear layer with a softmax activation function that maps the concatenation of terminal hidden states of the forward and backward LSTMs to K units, where K is the number of clusters. To assign an input example to a cluster, we first pass it through the encoder in $\mathcal{M}$ and feed the resulting encodings to the clustering model $\mathcal{C}$. The example is assigned to the cluster with the highest softmax output. We train this network via the procedure described in Section 3.3 using the Quora Pairs dataset. This network is also retrained after every 10 iterations to retain model-awareness.

The overall procedure is as follows. We divide the training dataset for the NLP task into 50 splits

[2] From a subset of the Brown Corpus (Burchfield, 1985), using splits from Martínez Alonso and Plank (2017)

[3] https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

of equal sizes and begin by training the model $\mathcal{M}$ on one of the randomly chosen splits (2% of the data). This model is then used to provide input to train the Siamese/Clustering network using the SICK/Quora Pairs dataset as described above. Next, at each iteration, we randomly pick one of the remaining splits and use an AL strategy to retrieve examples on which the model has low confidence. This is followed by clustering to extract the most representative examples using either spectral clustering (for Siamese based approach), or by directly using the output of the Integrated Clustering model. We average the results over five independent runs of the entire process with randomly chosen initial splits. See Appendix C for details about the choice of hyper-parameters used in our experiments.

4.4 Baselines

It is important to note that our proposed framework is not an AL strategy. Rather, it is an approach that further mitigates the redundancies in the existing AL strategies by working in conjunction with them. We validate our claims by comparing our approaches with three baselines that highlight the importance of various components.

Cosine : Clustering is done based on cosine similarity between last output encodings (corresponding to sentence length) from encoder in $\mathcal{M}$. Although this similarity computation is model-aware, it is simplistic and shows the benefit of using a more expressive similarity measure.

None : In this baseline, we use the AL strategy without applying Active^2 learning to remove redundant examples. This validates our claim about redundancy in examples chosen by AL strategies.

Random : No active learning is used and random examples are selected at each time.

4.5 Ablation Studies

We perform ablation studies to demonstrate the utility of model-awareness using these baselines:

Infersent : Clustering is done based on cosine similarity between sentence embeddings (Chen et al., 2015) obtained from a pre-trained InferSent model (Conneau et al., 2017). This similarity computation is not model-aware. This baseline shows the utility of model-aware similarity computation.

Task	Dataset	% of train data used to reach 99% of full-data F-Score	% less data required to reach 99% of full-data F-score
POS	CoNLL 2003	25%	16%
NER	CoNLL 2003	37%	3%
SEMTR	SEMCOR	35%	25%
CHUNK	CoNLL 2000	23%	11%

Table 2: Fraction of data used for reaching full dataset performance and the corresponding **absolute percentage reduction** in the data required over the None baseline that uses active learning strategy without the A^{2}L step for the best AL strategy (BALD in all cases).

Iso Siamese : To show that the Siamese network alone is not sufficient and model-awareness is needed, in this baseline, we train the Siamese network by directly using GloVe embeddings of the words as input rather than using output from model $\mathcal{M}$'s encoder. This similarity, which is not model-aware, is then used for clustering.

5 Results

Figure 1 compares the performance of our methods with baseline methods. It shows the test-set F-score on y-axis against percentage of training data used on x-axis for all AL strategies and one dataset per task. See Figures 4 and 5) in Appendix for additional results.

1. As shown in Figure 1, our approach consistently outperforms all baseline approaches on all chosen NLP tasks. Note that the one should look at how fast the performance increases as more training data is added and not just at the final performance as we are trying to evaluate the effect of adding new examples (Table 3).
2. Our ablation studies in Figure 2 show the utility of using model-aware similarity. See Figure 4 in Appendix for more experiments.
3. We match the performance obtained by training on full dataset using a smaller fraction of the data ($\approx \mathbf{3}-\mathbf{25}\%$ less data as compared to state-of-art AL strategies) (Table 2).
4. While comparing different AL strategies is not our motive, Figure 1 also demonstrates that one can achieve performance comparable to a complex AL strategy like BALD using simple AL

[4] We process the dataset to use only those sentences which are present in at least 5 other pairs. This leaves us with 16000 sets, each a source sentence and 5 samples (with both positive and negative labels). An additional 1000 sets were generated for evaluation.

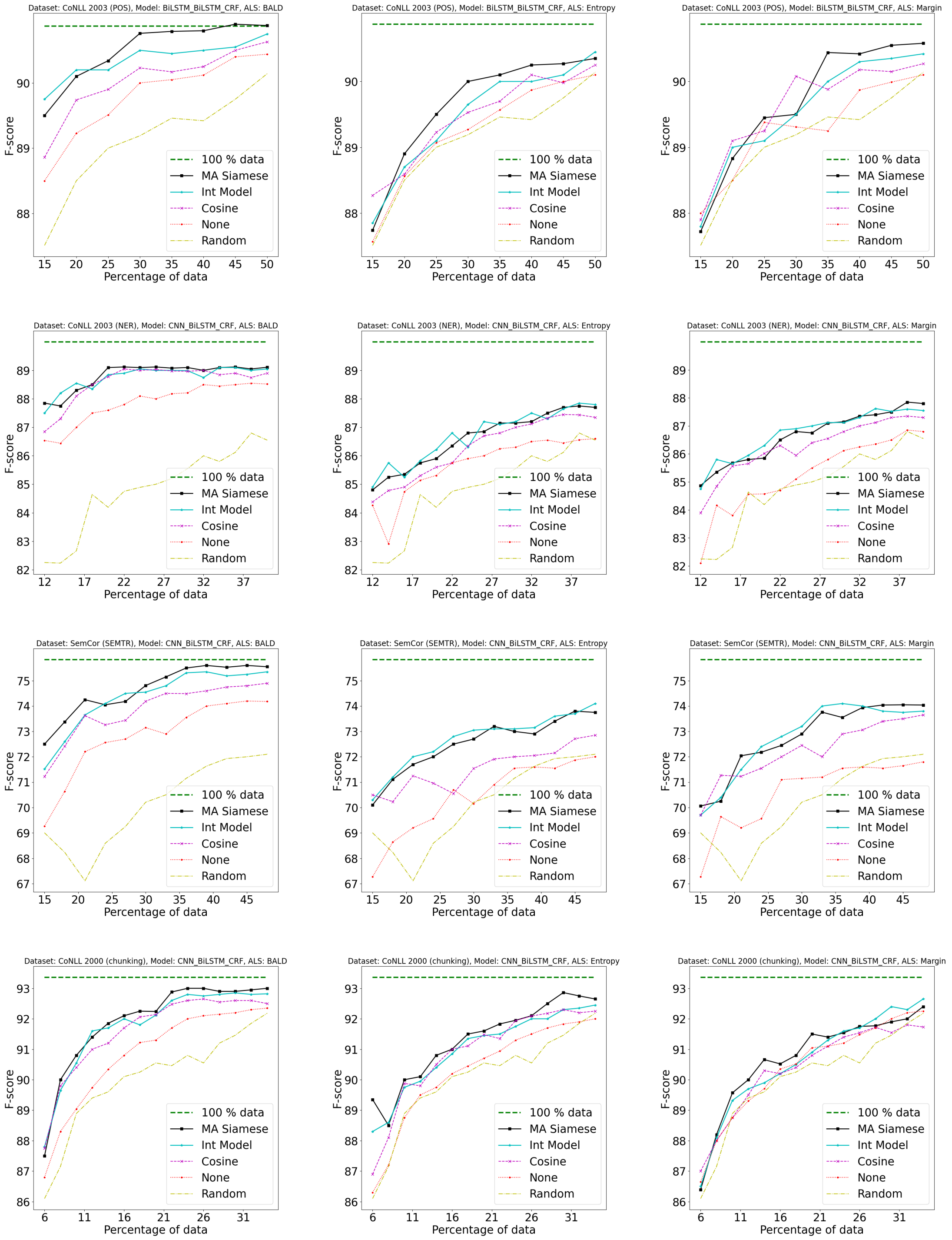

Figure 1: [Best viewed in color] Comparison of our approach (A^{2}L) with baseline approaches on different tasks using different active learning strategies. 1st row: POS, 2nd row: NER, 3rd row: SEMTR, 4th row: CHUNK. In each row, from left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description {**100% data** : full data performance, **A^{2}L (MA Siamese)** : Model Aware Siamese, **A^{2}L (Int Model)** : Integrated Clustering Model, **Cosine** : Cosine similarity, **None** : Active learning strategy without clustering step, **Random** : Random split (no active learning applied)}. See Section 4.4 for more details. All the results were obtained by averaging over 5 random splits. These plots have been magnified to highlight the regions of interest. For original plots, refer Fig 5 in Appendix.

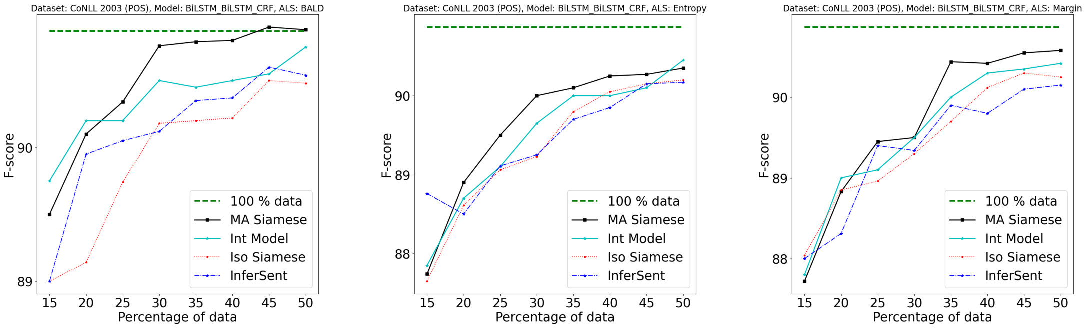

Figure 2: [Best viewed in color] Ablations studies on POS task using different active learning strategies. From left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description {**100% data** : full data performance, **A^{2}L (MA Siamese)** : Model Aware Siamese, **A^{2}L (Int Model)** : Integrated Clustering Model, **Iso Siamese** : Model isolated Siamese, **InferSent** : Cosine similarity based on InferSent encodings}. See Figure 4 in Appendix for experiments on other tasks. All the results were obtained by averaging over 5 splits.

% data **Setup**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**	**40%**	**45%**	**50%**
Iso Siamese	88.58	89.00	89.14	89.74	90.18	90.20	90.22	90.50	90.48
Cosine	88.34	88.86	89.74	89.90	90.23	90.17	90.25	90.50	90.63
InferSent	88.15	89.00	89.95	90.05	90.12	90.35	90.37	90.60	90.54
None (BALD)	88.58	88.50	89.23	89.51	90.00	90.05	90.12	90.40	90.44
Random (No ALS)	86.79	87.51	88.50	89.00	89.19	89.46	89.42	89.75	90.14
A^{2}L (MA Siamese)	**89.13**	89.50	90.10	**90.34**	**90.76**	**90.79**	**90.80**	90.70	**90.88**
A^{2}L (Int Model)	89.00	**89.75**	**90.20**	90.20	90.50	90.45	90.50	**90.75**	90.75

Table 3: Interpretation of the plot on the top left corner of Fig 5 (CoNLL 2003 (POS), BALD) in Appendix. The values in the cells are F-scores on the test set after training on the corresponding percentage of the data. It can be seen that with the increase in % labeled data, A^{2}L (MA Siamese) consistently performs better than other baselines.

strategies like Margin and Entropy based approach by using the proposed A^{2}L framework.

5. Our Siamese model (MA Siamese) performs slightly better than the Integrated Clustering model (Int Model) as the second approximates the first (at a lower computational cost).
6. Models usually require a tremendous amount of additional data for even a slight increase in F1 score, especially when the performance approaches the full-data F1 score (see Figure 5 in Appendix for details). Taking into account the wide applicability of our setup, for large datasets, even a 3% reduction would lead to significant reduction in annotation cost.

See Appendix A for a qualitative case study that demonstrates the problem of redundancy.

It should be noted that the reported improvement numbers are not relative with respect to any baseline but represent an absolute improvement. The magnitude of these figures is very significant in the context of similar performance improvements that have been reported in the literature.

6 Conclusion

In this paper, we proposed two methods that mitigate redundancy in existing AL strategies by using model-aware similarity computation. We empirically demonstrated that our proposed approaches consistently performs well across many tasks and AL strategies. We compared the performance of our approach with strong baselines to ensure that the role of each component is properly understood. Although, we focused on the sequence tagging problems, it would be interesting to apply the A^{2}L technique to other NLP tasks (also see Appendix B.2). It would also be interesting to further explore the role played by the auxiliary similarity datasets and/or develop approaches that do not rely on the availability of such datasets.

References

Pranjal Awasthi, Maria Florina Balcan, and Philip M. Long. 2017. The power of localization for efficiently learning linear separators with noise. *J. ACM*, 63(6):50:1–50:27.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 1613–1622. JMLR.org.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "siamese" time delay neural network. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, pages 737–744. Morgan-Kaufmann.

Robert Burchfield. 1985. Frequency analysis of english usage: Lexicon and grammar. by w. nelson francis and henry kuera with the assistance of andrew w. mackie. boston: Houghton mifflin. 1982. x + 561. *Journal of English Linguistics*, 18(1):64–70.

Yukun Chen, Thomas A. Lasko, Qiaozhu Mei, Joshua C. Denny, and Hua Xu. 2015. A study of active learning methods for named entity recognition in clinical text. *J. of Biomedical Informatics*, 58(C):11–18.

Vincent Claveau and Ewa Kijak. 2017. Strategies to select examples for active learning with conditional random fields. In *CICLing 2017 - 18th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 1–14.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. 2009. Analysis of perceptron-based active learning. *J. Mach. Learn. Res.*, 10:281–299.

Rosa Figueroa, Qing Zeng-Treitler, Long Ngo, Sergey Goryachev, and Eduardo Wiechmann. 2012. Active learning for clinical text classification: Is it better than random sampling? *Journal of the American Medical Informatics Association : JAMIA*, 19:809–16.

Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. 1997. Selective sampling using the query by committee algorithm. *Mach. Learn.*, 28(2-3):133–168.

Yarin Gal and Zoubin Ghahramani. 2016a. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1050–1059. JMLR.org.

Yarin Gal and Zoubin Ghahramani. 2016b. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Mt Lengyel. 2011. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Shari Landes and Claudia Leacock. 1998. Building a semantic concordance of english. *WordNet: An Electronic Lexical Database*.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 44–53. Association for Computational Linguistics.

Andrew McCallum and Kamal Nigam. 1998. Employing em and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 350–358.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2786–2792.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 849–856. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew S. Ryan and Graham R. Nudd. 1993. The viterbi algorithm. Technical report, -.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *Proceeding of the Computational Natural Language Learning (CoNLL)*.

Fei Sha and Lawrence K. Saul. 2007. Large margin hidden markov models for automatic speech recognition. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1249–1256. Curran Associates, Inc.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *6th International Conference on Learning Representations*.

Aditya Siddhant and Zachary C Lipton. 2018. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Cynthia A. Thompson, Mary Elaine Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML 99, page 406414, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

A Understanding Redundancy and Model Aware Similarity

To convey the notion of redundancy and the idea of model aware similarity, in this section, we examine some example sentences that were deemed similar by the model aware Siamese in our proposed approach. To obtain these examples, we followed the training procedure outlined in Section 4.3 for the NER task on CoNLL 2003 dataset. After the model had seen roughly 10% of the data, we collected examples that were: **(i)** selected by the AL strategy (BALD) as examples on which the model has low confidence, and **(ii)** grouped by the clustering procedure in the same cluster based on model aware Siamese similarity scores. We present two sentences each from some randomly chosen clusters below:

1. **Cluster 1**:
   - Russian **(B-MISC)** double Olympic **(B-MISC)** swimming champion Alexander **(B-PER)** Popov **(I-PER)** was in a serious condition on Monday after being stabbed on a Moscow **(B-LOC)** street.
   - Vitaly **(B-PER)** Smirnov **(I-PER)**, president of the Russian **(B-MISC)** National **(I-MISC)** Olympic **(I-MISC)** Committee **(I-MISC)**, said President Boris **(B-PER)** Yeltsin **(I-PER)** had given the swimmer Russia's **(B-LOC)** top award for his Olympic **(B-MISC)** performance.
2. **Cluster 2**:
   - The newspaper said the Central **(B-ORG)** Bank **(I-ORG)** special administration of Banespa **(B-ORG)** ends in December 30 and after that the bank has to be liquidated or turned into a federal bank since there are no conditions to return Banespa **(B-ORG)** to Sao **(B-LOC)** Paulo **(I-LOC)** state government.
   - The newspaper said Bamerindus **(B-ORG)** has sent to the Central **(B-ORG)** Bank **(I-ORG)** a proposal for restructuring combined with a request for a 90-day credit line, paying four percent a year plus the Basic Interest Rate of the Central **(B-ORG)** Bank **(I-ORG)** (TBC **(B-ORG)**).

Ground truth tags have been reported alongside the words, except for the words that belong to the "Other" class. For the sake of comparison, we also provide examples from two clusters that were obtained by using the cosine similarity metric on the InferSent embedding (Infersent baseline described in Section 4.4). As in the previous case, these examples have been selected by the AL strategy (BALD) for the same task and dataset as before. Note that similarity computation is not model aware in this case.

1. **Cluster 1**:
   - "His condition is serious," said Rimma **(B-PER)** Maslova **(I-PER)**, deputy chief doctor of Hospital **(B-LOC)** No **(I-LOC)** 31 **(I-LOC)** in the Russian **(B-MISC)** capital.
   - Popov **(B-PER)** told NTV **(B-ORG)** television on Sunday he was in no danger and promised he would be back in the pool shortly.
2. **Cluster 2**:
   - MOTORCYCLING - JAPANESE **(B-MISC)** WIN BOTH ROUND NINE SUPERBIKE RACES.
   - Honda's **(B-ORG)** Takeda **(B-PER)** was pursued past Corser **(B-PER)** by the Yamaha **(B-ORG)** duo of Noriyuki **(B-PER)** Haga **(I-PER)** and Wataru **(B-PER)** Yoshikawa **(I-PER)** with Haga **(B-PER)** briefly taking the lead in the final chicane on the last lap.

As expected, when cosine similarity is used, sentences that have roughly similar content have been assigned to the same cluster. However, when model aware similarity is used, in addition to having similar content, the sentences also have the similar tagging structure. As the InferSent based similarity is agnostic of the downstream task, it cannot predict similarity between sentences based on the downstream task unlike the model aware Siamese approach. However, for the NER task, it is sensible to eliminate sentences having similar tagging structure, as they are redundant as far as the learning on the downstream task is concerned.

This example not only supports our claim that AL strategies choose redundant examples, it also highlights the utility of using model aware similarity computation.

B Additional Remarks

In this section we make a number of additional remarks about the proposed approach.

B.1 What is the significance of our work?

Obtaining labeled data is both time consuming and costly. Active learning is employed to minimize the labeling effort, however, as we point out in Section 1, existing techniques may select redundant examples for manual annotation. Due to this redundancy, there is a scope for improvement in the performance of active learning strategies and our proposed approach fills this gap. Since we demonstrate that our method is compatible with many active learning strategies and deep learning models that are currently in use (also see Section B.2), it can be applied in a wide range of contexts and is likely to be useful for many sub-communities within the domain of natural language processing without adding significant complexity to the existing systems.

B.2 Can the approach be applied to other NLP tasks?

Active^2 learning works in conjunction with an active learning strategy. Once the active learning strategy has selected the examples to be labeled, our approach only expects that the underlying deep learning model is accessible to the Siamese network for obtaining its input. Thus, as long as an active learning strategy can be applied in a given context, our approach is also applicable. For example, our approach may be used for tasks like machine translation, sentiment analysis and so on. In this paper we focus on the sequence tagging tasks for two reasons: **(i)** we believe that obtaining labeled data for sequence tagging tasks is especially challenging; and **(ii)** the chosen sequence tagging tasks form the backbone for many practical NLP systems.

B.3 How do we validate our claim regarding the sub-optimality of standard AL strategies due to redundancy?

The comparison of our approach with None baseline suggests that performance comparable to the state-of-art can be achieved by using fewer labels if one incorporates the second step which eliminates allegedly redundant examples even when every other aspect of training is exactly the same (same model, AL strategy and dataset). Thus, we can say that the discarded examples were of no additional help for the model and hence were redundant. Avoiding annotation of such samples saves time and brings down both computational and annotation costs. This can especially be effective in, for instance, the medical domain where high expertise is required.

C Hyper-parameters and other Implementation Details

Active Learning strategy	
threshold (Margin)	15
threshold (Entropy)	40
threshold (BALD)	0.2
dropout (BALD)	0.5
number of forward passes (BALD)	51
Sequence tagging model	
CNN filter sizes	[2,3]
training batch size	12
splits of train data	50
number of train epochs	16
dimension of character embedding	100
learning rate (Adam)	0.005
learning rate decay	0.9
Siamese encoder	
training batch size	48
number of train epochs	41
train/dev split	0.8
learning rate (Adam)	1e-5
period (of retrain)	10
Clustering	
Number of clusters	20
Training	
Batch size	12

Similar hyper-parameter values work across all the tasks and hence the same values were used for all experiments and these values were determined using the validation set of CoNLL 2003 dataset for NER task.

We use two different sequence tagging architectures: CNN-BiLSTM-CRF model (CNN for character-level encoding and BiLSTM for word-level encoding) and a BiLSTM-BiLSTM-CRF model (Lample et al., 2016) (BiLSTM for both character-level and word-level encoding). The CNN-BiLSTM-CRF architecture is a light-weight variant of the model proposed in (Siddhant and Lipton, 2018), having one layer in CNN encoder with two filters of sizes 2 and 3, followed by a max pool, as opposed to three layers in the original setup. This modification was found to improve the re-

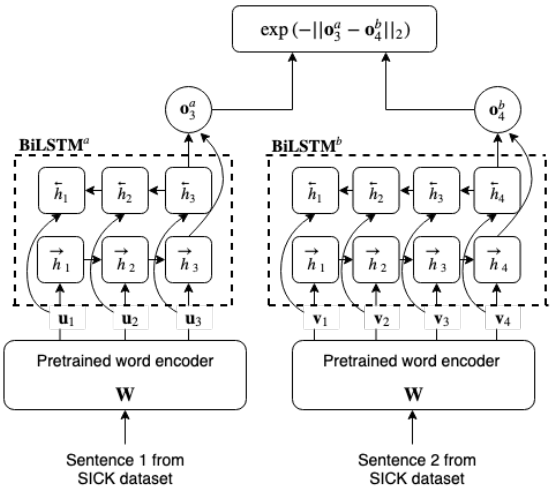

Figure 3: Modeling similarity using the Siamese encoder (enclosed by dotted lines). A pair of sentences from SICK dataset is fed to the pretrained sequence tagging model. The output of the word encoder is then passed to the Siamese encoder. Last hidden state of the Siamese encoder, corresponding to the sequence length of the sentence, is used for assigning a similarity score to the pair.

sults. We use glove embeddings (Pennington et al., 2014) for all datasets. We apply normal dropout in the character encoder as opposed to the use of recurrent dropout (Gal and Ghahramani, 2016b) in the word encoder of model presented in (Siddhant and Lipton, 2018) owing to an improvement in performance. For numerical stability, we use log probabilities and, thus, the value for margin based AL strategy's threshold is outside the interval $[0, 1]$. We use the spectral clustering (Ng et al., 2002) algorithm to cluster the sentences chosen by AL strategy. We chose two representative examples from each cluster.

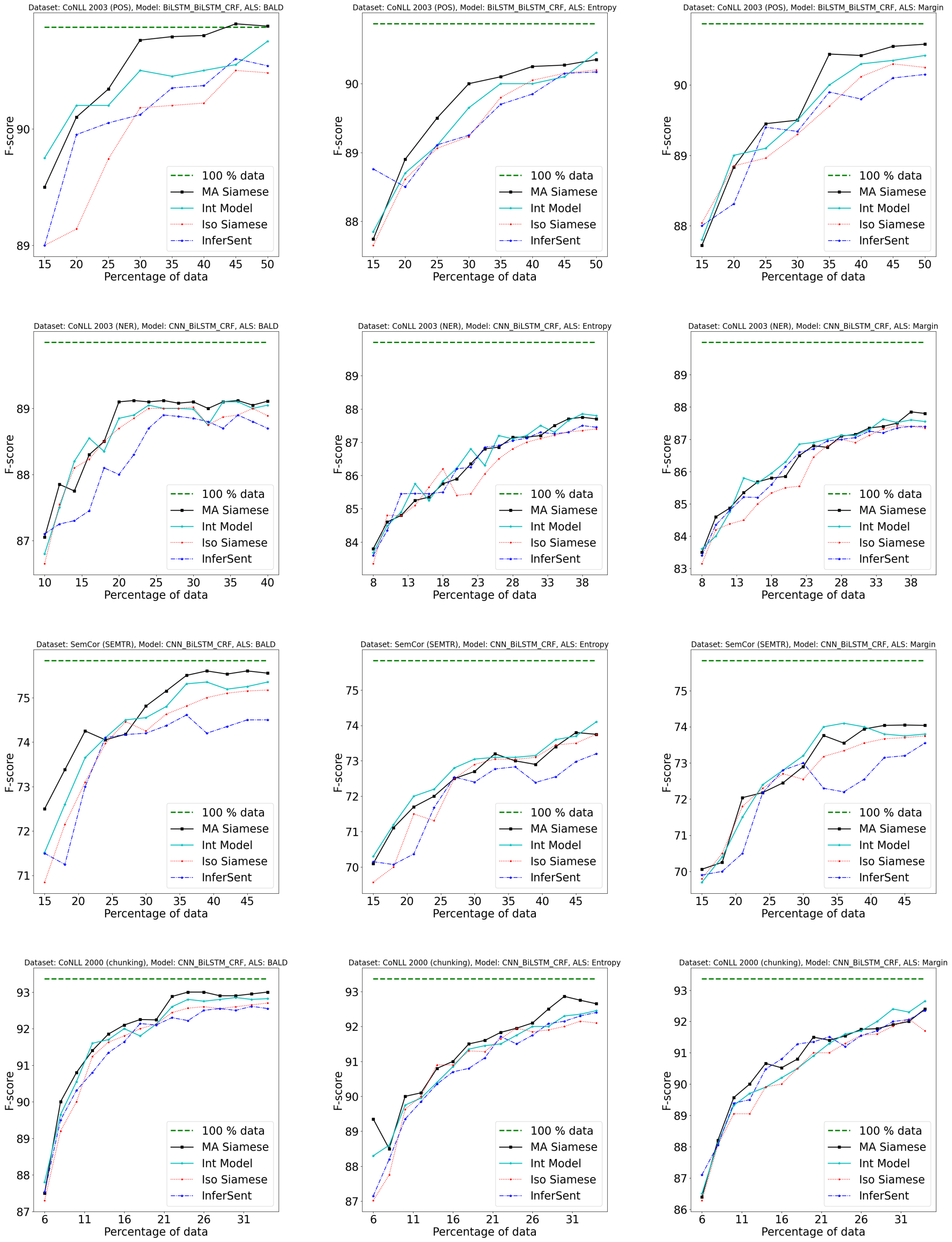

Figure 4: [Best viewed in color] Ablations studies on different tasks using different active learning strategies. 1^{st} row: POS, 2^{nd} row: NER, 3^{rd} row: SEMTR, 4^{th} row: CHUNK. In each row, from left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description $\{$**100%** data : full data performance, **A^{2}L (MA Siamese)** : Model Aware Siamese, **A^{2}L (Int Model)** : Integrated Clustering Model, **Iso Siamese** : Model isolated Siamese, **InferSent** : Cosine similarity based on InferSent encodings$\}$. See Section 4.5 for more details. All results were obtained by averaging over 5 random splits.

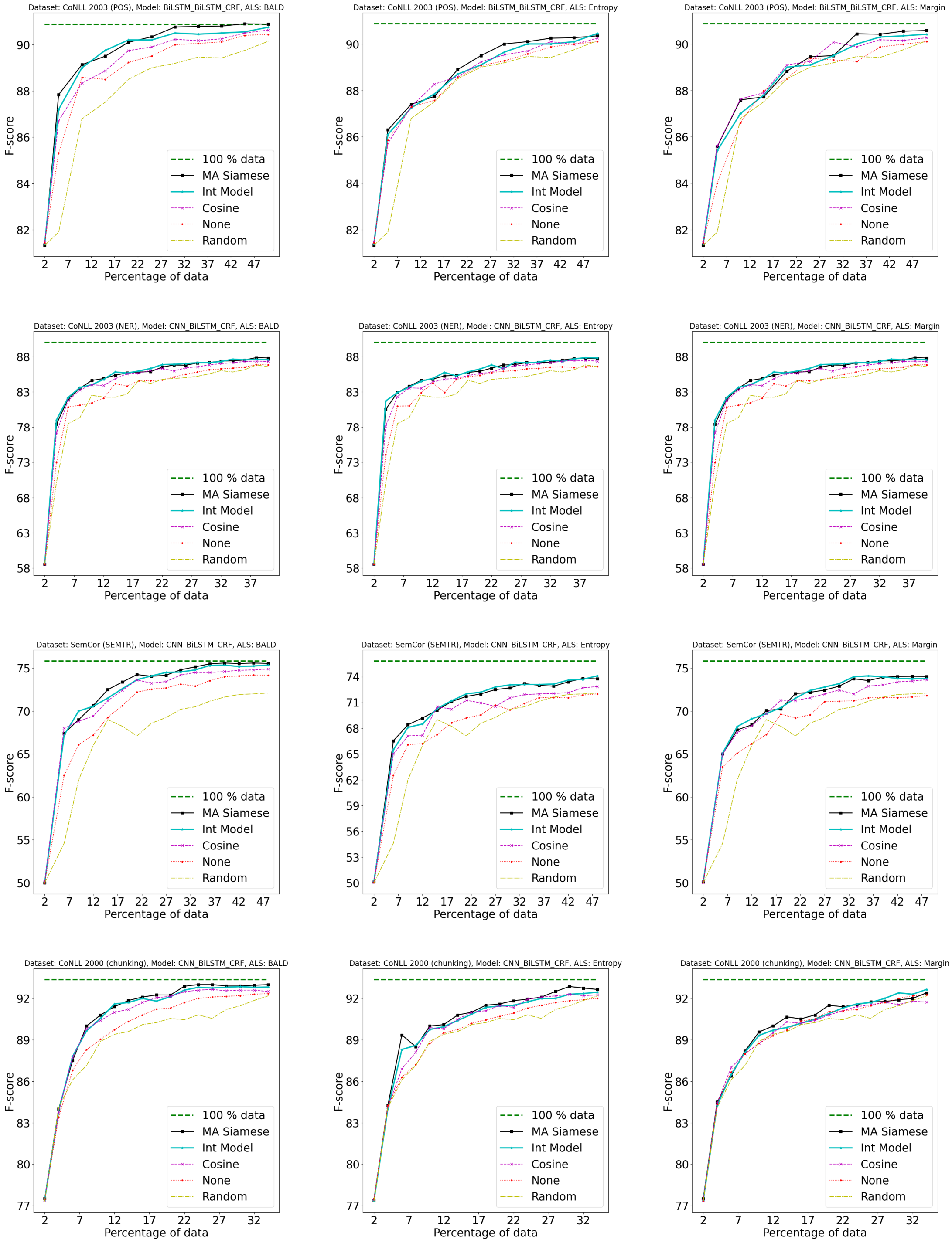

Figure 5: [Best viewed in color] Comparison of our approach (A^{2}L) with baseline approaches on different tasks using different active learning strategies. 1^{st} row: POS, 2^{nd} row: NER, 3^{rd} row: SEMTR, 4^{th} row: CHUNK. In each row, from left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description {**100%** data : full data performance, **A^{2}L (MA Siamese)** : Model Aware Siamese, **A^{2}L (Int Model)** : Integrated Clustering Model, **Cosine** : Cosine similarity, **None** : Active learning strategy without clustering step, **Random** : Random split (no active learning applied)}. See Section 4.4 for more details. All the results were obtained by averaging over 5 random splits.