

# Active<sup>2</sup> Learning: Actively reducing redundancies in Active Learning methods for Sequence Tagging and Machine Translation

Rishi Hazra, Parag Dutta, Shubham Gupta,  
Mohammed Abdul Qaathir, Ambedkar Dukkipati

{rishihazra, paragdutta, shubhamg,  
mohammedq, ambedkar}@iisc.ac.in

Indian Institute of Science, Bangalore

## 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<sup>2</sup> Learning (A<sup>2</sup>L), actively adapts to the deep learning model being trained to eliminate such redundant examples chosen by an AL strategy. We show that A<sup>2</sup>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  $\approx 3 - 25\%$  on an absolute scale on multiple NLP tasks while achieving the same performance with virtually no additional computation overhead.

## 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 sequence tagging (NER, POS) and Neural Machine Translation (NMT) 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).

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. This paper addresses this problem by proposing a new approach called A<sup>2</sup>L (read as active-squared learning) that further reduces the redundancies of existing AL strategies.

Any approach for eliminating redundant examples must have the following qualities: **(i)** The redundancy should be evaluated in the context of the trained model. **(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 model’s perspective. If the model is confused about one such example, it will likely be confused about all of them. We refer to a similarity measure that is computed in the context of a model as a *model-aware similarity* (Section 3.1).

<sup>0</sup>In proceedings of NAACL-HLT 2021

**Contributions:** (i) We propose a Siamese twin- (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 existing 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 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; Haffari et al., 2009; Bloodgood and Callison-Burch, 2010; Blundell et al., 2015; Gal and Ghahramani, 2016a), however, since they choose the examples independently, the problem of redundancy (Section 1) applies to all.

We experiment with various NLP tasks like named entity recognition (NER) (Nadeau and Sekine, 2007), part-of-speech tagging (POS) (Marcus et al., 1993), neural machine translation (NMT) (Hutchins, 2004; Nepveu et al., 2004; Bahdanau et al., 2014; Cho et al., 2014; Sutskever et al., 2014; Ortiz-Martínez, 2016) and so on (Landes et al., 1998; Tjong Kim Sang and Buchholz, 2000). The tasks chosen by us form the backbone of many practical problems and are known to be computationally expensive during both training and inference. Many deep learning models have recently advanced the state-of-art for these tasks (Bahdanau

et al., 2014; Lample et al., 2016; Siddhant and Lipton, 2018). Our proposed approach is compatible with any NLP model, provided it supports the usage of an AL strategy.

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 pre-calculate pairwise similarity between examples. We instead argue in 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 Inferred embeddings (Conneau et al., 2017).

## 3 Proposed Approaches

We use  $\mathcal{M}$  to denote the model being trained for a given task.  $\mathcal{M}$  has a module called encoder for encoding the input sentences. For instance, the encoder in  $\mathcal{M}$  may be modeled by an LSTM (Hochreiter and Schmidhuber, 1997).

### 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 Inferred 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 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. 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.

### 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 out-

---

**Algorithm 1: Active<sup>2</sup> Learning**

---

**Data:**  $\mathcal{D}_1$ : task dataset;  
           $\mathcal{D}_2$ : auxiliary similarity dataset  
**Input:**  $\mathcal{D} \leftarrow 2\%$  of dataset  $\mathcal{D}_1$ ;  
           $\mathcal{D} \leftarrow \mathcal{D}_1 - \mathcal{D}$ ; // unlabeled data  
**Output:** Labeled data  
**Initialization:**  $\mathcal{D}$ ;  
 $\mathcal{M} \leftarrow \text{TRAIN}(\mathcal{D})$ ;  
 $\mathcal{M}_{A^2L} \leftarrow \text{TRAIN}(\mathcal{M}(\mathcal{D}_2))$ ;  
**for**  $i \leftarrow 1$  **to**  $l$  **do**  
     $\mathcal{S} \leftarrow \mathcal{AL}(\mathcal{D})$ ; // top 2% confused  
        samples  
    **if** // Model-Aware Siamese  
    **then**  
        **for each pair**  $(s_m, s_n)$  **in**  $\mathcal{S}$  **do**  
             $\mathbf{S}[m, n] \leftarrow \mathcal{M}_{A^2L}(s_m, s_n)$ ;  
             $\mathcal{R} \leftarrow \text{CLUSTER}(\mathbf{S})$ ;  
    **else**  
        // Integrated Clustering  
         $\mathcal{R} \leftarrow \mathcal{M}_{A^2L}(\mathcal{S})$ ;  
     $\mathcal{R} \leftarrow \text{ANNOTATE}(\mathcal{R})$ ;  
     $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{R}$ ;  
     $\mathcal{M} \leftarrow \text{RETRAIN}(\mathcal{D})$

---

put of model  $\mathcal{M}$ 's encoder. The outputs of the Siamese encoder are used for computing the similarity between each pair of examples  $a$  and  $b$  as:

$$\text{sim}(a, b) = \exp(-\|\mathbf{o}^a - \mathbf{o}^b\|_2), \quad (1)$$

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  $S_{ab} = \text{sim}(a, b)$ . 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 the 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 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 the 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 (see Figure 1).

### 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 (see Figure 1) 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  $c$ ). We experimented with both SICK and Quora Pairs dataset<sup>3</sup>. 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  $a$  is treated as the ground-truth class for  $b$ . Minimizing the objective given below maximizes the probability of  $b$  belonging to its ground truth class while minimizing the probability of  $c$  belonging to the same class:

$$\begin{aligned} \mathcal{L}(a, b, c) = & -\lambda_1 \log p_{i_a}^b - \lambda_2 \log(1 - p_{i_a}^c) \\ & + \lambda_3 \sum_{k=1}^K p_k^b \log p_k^b. \end{aligned} \quad (2)$$

Here  $\lambda_1$ ,  $\lambda_2$ , and  $\lambda_3$  are user-specified hyperparameters,  $p_j^x$  is the softmax output of the  $j^{\text{th}}$  unit for example  $x$ ,  $j = 1, 2, \dots, K$ ,  $x = a, b, c$ , and  $i_a = \arg \max_{j \in \{1, 2, \dots, K\}} p_j^a$ . 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<sup>2</sup> Learning.

## 4 Experiments

We establish the effectiveness of our approaches by demonstrating that they: **(i)** work well across a variety of NLP tasks and models, **(ii)** are compatible with several popular AL strategies, and **(iii)** further reduce the data requirements of existing AL strategies, while achieving the same performance. In particular, we experiment<sup>1</sup> with two broad categories of NLP tasks: (a) Sequence Tagging (b) Neural Machine Translation. Table 1 lists these tasks and information about the corresponding datasets (including the two auxiliary datasets for training the Siamese network (Section 3.2)) used in our experiments. We begin by describing the AL strategies for the two kinds of NLP tasks.

### 4.1 Active Learning Strategies for Sequence Tagging

**Margin-based strategy:** Let  $s(\mathbf{y}) = P_{\theta}(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x})$  be the score assigned by a model  $\mathcal{M}$  with parameters  $\theta$  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.:

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

where,  $\mathbf{y}_{\text{max}} = \arg \max_{\mathbf{y}} s(\mathbf{y})$ . The strategy selects examples for which  $M_{\text{margin}} \leq \tau_1$ , where  $\tau_1$

<sup>1</sup>Codes for the experiments are available at the following github link: <https://github.com/parag1604/A2L>.

is a hyper-parameter. We use Viterbi’s algorithm (Ryan and Nudd, 1993) to compute the scores  $s(\mathbf{y})$ .

**Entropy-based 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  $\bar{s}_j = \max_{o \in \mathcal{O}} P_{\theta}(y_j = o | \mathbf{X} = \mathbf{x})$  to be the probability of the most likely output for the  $j^{\text{th}}$  token in  $\mathbf{x}$ . Here  $\mathcal{O}$  is set of all possible outputs and  $y_j$  is the output corresponding to the  $j^{\text{th}}$  token in  $\mathbf{x}$ . We define the *normalized entropy score* as:

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

A length normalization  $n(\mathbf{x})$  is added to avoid bias due to the example length as it may be undesirable to annotate longer length examples (Claveau and Kijak, 2017). The strategy selects examples with  $M_{\text{entropy}} \geq \tau_2$ , where  $\tau_2$  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  $t^{\text{th}}$  forward pass, and let  $N$  be the number of forward passes with a fixed dropout rate, then:

$$M_{\text{bald}} = 1 - \frac{\text{count}(\text{mode}(\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(N)}))}{N}. \quad (5)$$

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

### 4.2 Active Learning Strategies for Neural Machine Translation<sup>2</sup>

**Least Confidence (LC)** This strategy estimates the uncertainty of a trained model on a source sentence  $\mathbf{x}$  by calculating the conditional probability of the prediction  $\hat{\mathbf{y}}$  conditioned on the source sentence (Lewis and Catlett, 1994).

$$M_{\text{LC}} = \frac{1}{n(\hat{\mathbf{y}})} \log \mathbf{P}(\hat{\mathbf{y}} | \mathbf{x}) \quad (6)$$

<sup>2</sup>The AL strategies for NMT are ranking-based techniques, so we select the top 50% of the candidates after sorting them in ascending (6,7) or descending (9) order.

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 <sup>2</sup>	13851 / 4696	This section prevents the military departments 0 Mental Agentive 0 0 Object
NMT	Europarl (en → es)	100000 / 29155	(1) that is almost a personal record for me this autumn ! (2) es la mejor marca que he alcanzado este otoño .
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 <sup>3</sup>	16000 / 1000 (sets) <sup>5</sup>	(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<sup>3</sup>, Europarl (Koehn, 2005), SICK (Marelli et al., 2014), Quora Pairs<sup>4</sup>.

A length normalization of  $n(\hat{y})$  (length of the predicted translation  $\hat{y}$ ) is added.

**Coverage Sampling (CS)** A translation model is said to *cover* the source sentence if it translates all of its tokens. Coverage is estimated by mapping a particular source token to its appropriate target token, without which the model may suffer from under-translation or over-translation issues (Tu et al., 2016). Peris and Casacuberta (2018) proposed to use translation coverage as a measure of uncertainty by:

$$M_{CS} = \frac{\sum_{j=1}^{n(\mathbf{x})} \log(\min(\sum_{i=1}^{n(\hat{y})} \alpha_{i,j}, 1))}{n(\mathbf{x})} \quad (7)$$

Here  $\alpha_{i,j}$  denotes the attention probability calculated by the model for the  $j^{th}$  source word in predicting the  $i^{th}$  target word. It can be noted that the coverage score will be 0 for samples for which the model almost fully covers the source sentences.

**Attention Distraction Sampling (ADS)** Peris and Casacuberta (2018) claimed that in translating an uncertain sample, the model’s attention mechanism would be *distracted* (dispersed throughout the sentence). Such samples yield attention probability distribution with light tails (e.g., uniform distribution), which can be obtained by taking the Kurtosis of the attention weights for each target token  $\mathbf{y}_i$ .

$$\text{Kurt}(\mathbf{y}_i) = \frac{\frac{1}{n(\mathbf{x})} \sum_{j=1}^{n(\mathbf{x})} (\alpha_{i,j} - \frac{1}{n(\mathbf{x})})^4}{(\frac{1}{n(\mathbf{x})} \sum_{j=1}^{n(\mathbf{x})} \alpha_{i,j} - \frac{1}{n(\mathbf{x})})^2} \quad (8)$$

where  $\frac{1}{n(\mathbf{x})}$  is the mean of the distribution of the attention weights (for a target word) over the source

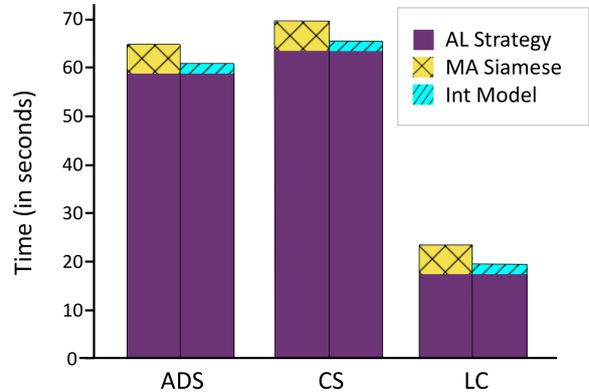


Figure 1: Comparison of time taken for one data selection step in NMT task by the Model Aware (MA) Siamese and Integrated Clustering (Int) Model across different ALS. It can be observed that A<sup>2</sup>L adds a negligible overhead ( $\approx \frac{1}{12}$  of the time taken for ALS) to the overall process.

words. The kurtosis value will be lower for distributions with light tails, so the average of the negative kurtosis values for all words in the target sentence is used as the distraction score.

$$M_{ADS} = \frac{\sum_{i=1}^{n(\mathbf{y})} -\text{Kurt}(\mathbf{y}_i)}{n(\mathbf{y})} \quad (9)$$

### 4.3 Details about Training

For sequence tagging, we use two kinds of architectures: CNN-BiLSTM-CRF model (CNN for character-level encoding and BiLSTM for word-level encoding) and a BiLSTM-BiLSTM-CRF

<sup>3</sup>From a subset of the Brown Corpus (Burchfield, 1985), using splits from Martínez Alonso and Plank (2017)

<sup>4</sup><https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs>

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<sup>2</sup>L step for the best AL strategy (BALD in all cases). Refer Fig 8 in Appendix for CHUNK plots.

model (BiLSTM for both character-level and word-level encoding) (Lample et al. (2016); Siddhant and Lipton (2018)). For the translation task, we use LSTM based encoder-decoder architecture with Bahdanau attention (Bahdanau et al., 2014). These models were chosen for their performance and ease of implementation.

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 the 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 by retraining the Siamese 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. This network is also retrained after every 10 iterations to retain model-awareness.

The initial data splits used for training the model

$\mathcal{M}$  were set at 2% of randomly sampled data for Sequence Tagging (20% for NMT). These are in accordance with the splitting techniques used in the existing literature on AL (Siddhant and Lipton, 2018; Liu et al., 2018). The model is then used to provide input to train the Siamese/Clustering network using the SICK/Quora Pairs. At each iteration, we gradually add another 2% of data for sequence tagging (5% for NMT) by retrieving low confidence examples using an AL strategy, followed by clustering to extract the most representative examples. We average the results over five independent runs with randomly chosen initial splits. [Hyperparameters details in Appendix B].

#### 4.4 Baselines

We claim that A<sup>2</sup>L 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<sup>2</sup> 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 and shows the utility of model-aware similarity computation.

**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

<sup>5</sup>We process the dataset to use only those sentences which are present in at least 5 other pairs. We retrieve 16000 sets, each with a source sentence and 5 other samples (comprising both positive and negative labels). An additional 1000 sets were generated for evaluation.

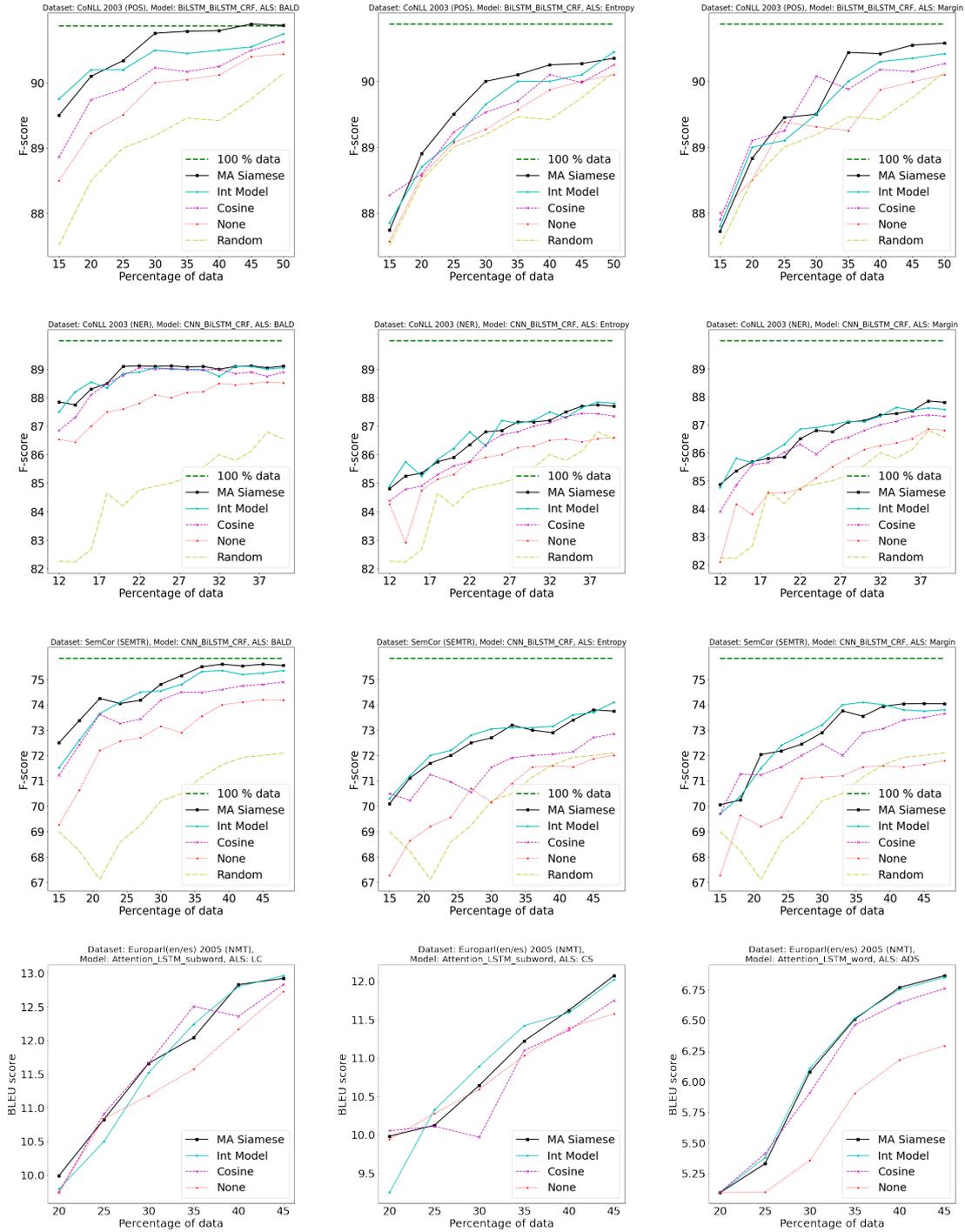


Figure 2: [Best viewed in color] Comparison of our approach ( $A^2L$ ) with baseline approaches on different tasks using different active learning strategies.  $1^{st}$  row: POS (error bound at convergence:  $\pm 0.05$ ),  $2^{nd}$  row: NER ( $\pm 0.08$ ),  $3^{rd}$  row: SEMTR ( $\pm 0.09$ ),  $4^{th}$  row: NMT ( $\pm 0.04$ ). In the first three rows, from left to right, the three columns represent BALD, Entropy, and Margin AL strategies.  $4^{th}$  row represents AL strategies for NMT, from left to right (LC: Least Confidence, CS: Coverage Sampling, ADS: Attention Distraction Sampling) : Legend Description {100% data: full data performance,  $A^2L$  (MA Siamese) : Model Aware Siamese,  $A^2L$  (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 about the baselines. 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 to Fig 8 in the Appendix.

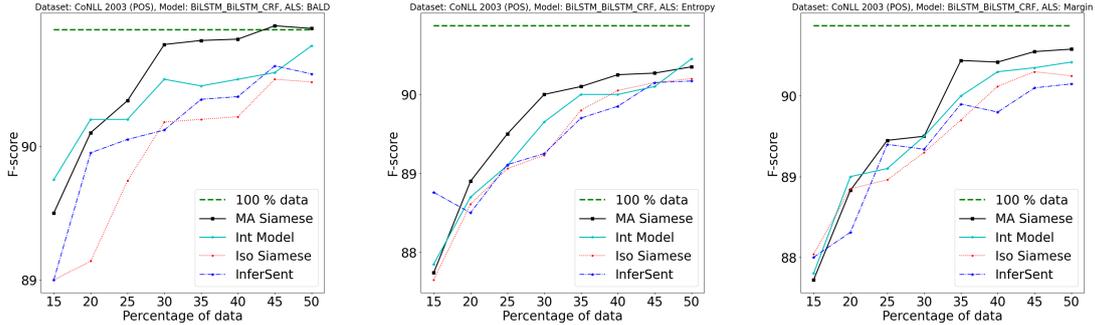


Figure 3: [Best viewed in color] Ablations studies on POS task using different active learning strategies. From left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description {**100%** data : full data performance,  $A^2L$  (MA Siamese) : Model Aware Siamese,  $A^2L$  (Int Model) : Integrated Clustering Model, Iso Siamese : Model isolated Siamese, InferSent : Cosine similarity based on InferSent encodings}. See Figure 7 in Appendix for experiments on other tasks. All the results were obtained by averaging over 5 splits.

words as input rather than using output from the model  $\mathcal{M}$ 's encoder. This similarity, which is not model-aware, is then used for clustering.

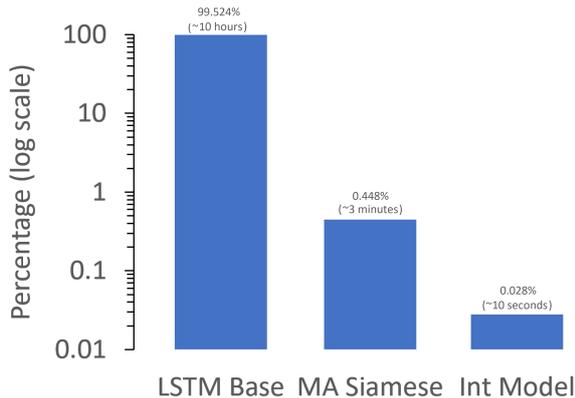


Figure 4: Comparison between various components of our approach in terms of the time required by them per training epoch for NMT. LSTM (Base) encoder-decoder translation model ( $\approx 10$  hours), Model Aware (MA) Siamese ( $\approx 3$  minutes) and Integrated Clustering (Int) Model ( $\approx 10$  seconds). It can be observed that  $A^2L$  adds a negligible overhead to the overall training time ( $\approx 0.45\%$  of the time taken by the base model).

## 5 Results

Figure 2 compares the performance of our methods with baselines. It shows the test-set metric on the  $y$ -axis against the percentage of training data used on the  $x$ -axis for all tasks. See Figures 7 and 8 in the Appendix for additional results.

1. As shown in Figure 2, our approach consistently outperforms all baselines on chosen tasks. Note that one should observe how fast the performance increases with the addition of training

data (and not just the final performance) as we are trying to evaluate the effect of adding new examples. Our ablation studies in Figure 3 show the utility of using model-aware similarity.

2. In sequence tagging, we match the performance obtained by training on the full dataset using only a smaller fraction of the data (**3 – 25%** less data as compared to state-of-art AL strategies) (Table 2). On a large dataset in NMT task (Europarl),  $A^2L$  takes  $\approx 4300$  sentences fewer than the Least Confidence AL strategy to reach a Bleu score of 12.
3. While comparing different AL strategies is not our motive, Figure 2 also demonstrates that one can achieve performance comparable to a complex AL strategy like BALD, using simple AL strategies like margin and entropy, by using the proposed  $A^2L$  framework.
4. Additionally, from Figure 1, it can be observed that for one step of data selection: (i) The proposed MA Siamese model adds minimal overhead to the overall AL pipeline since it takes less than 5 additional seconds ( $\approx \frac{1}{12}$  of the time taken for ALS); (ii) By approximating the clustering step, Integrated Clustering (Int) Model further reduces the overhead down to 2 seconds. However, owing to this approximation, MA Siamese is observed to perform slightly better than the Int Model (Fig 3). A comparison of training time for various stages of the  $A^2L$  pipeline is provided in Figure 4.

We wish to state that our approach should be evaluated not in terms of the gain in the F1 score but

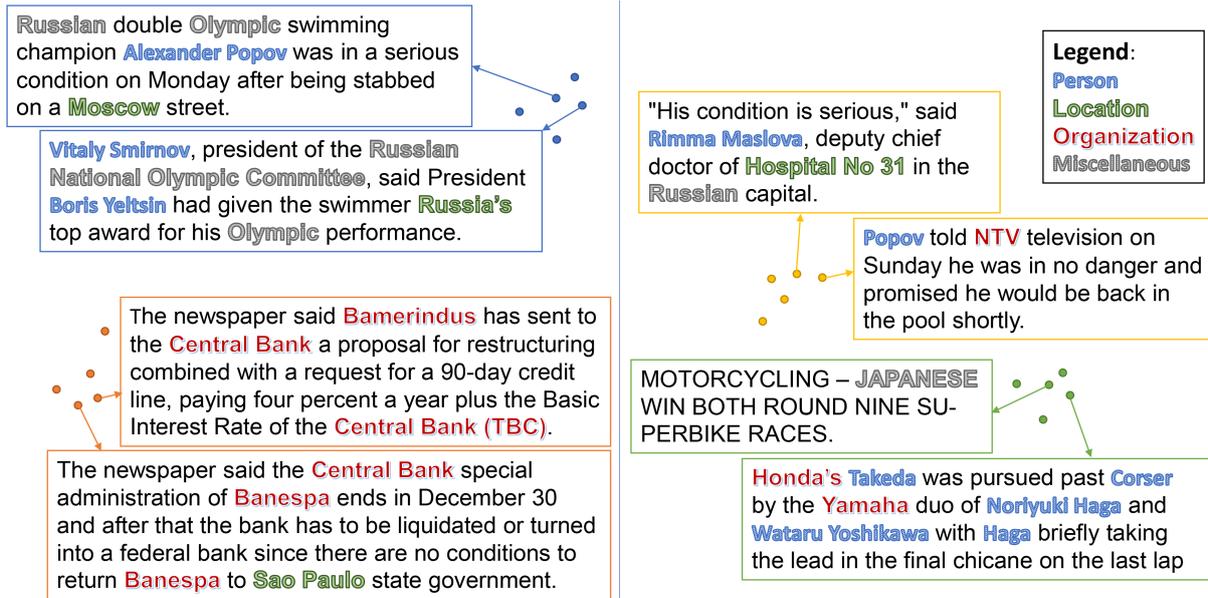


Figure 5: [Best viewed in color] Qualitative case study to convey the notion of redundancy and the model aware similarity. We compare the examples grouped in the same cluster using (i) MA similarity scores [left] and (ii) Cosine similarity scores on the InferSent embedding (InferSent baseline described in Section 4.4) [Right]. 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 a similar tagging structure. It is sensible to eliminate sentences having similar tagging structures, as they are redundant. [Dataset: CoNLL 2003 dataset, AL strategy: BALD, Data usage: 10% of the total data]

No. of examples selected by AL Strategy	Spectral (examples processed per second)	Integrated (examples processed per second)
5000	491.64	2702.70
10000	248.08	2557.54
15000	163.10	2508.36
20000	122.89	2493.76

Table 3: Number of samples processed per second by the Spectral Clustering (MA Siamese) compared to the Integrated Clustering (Int Model) methods.

in terms of the reduction in data required to achieve the same (3-25 % on multiple datasets). More importantly, this improvement comes at a negligible computation overhead cost. The reported improvements are not relative with respect to any baseline but represent an absolute value and are very significant in the context of similar performance improvements reported in the literature. In Figure 5, we provide a qualitative case study that demonstrates the problem of redundancy.

## 6 Conclusion

This paper shows that one can further reduce the data requirements of Active Learning strategies

by proposing a new method,  $A^2L$ , which uses a model-aware-similarity computation. We empirically demonstrated that our proposed approaches consistently perform 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.

## Acknowledgement

This work was funded by British Telecom India Research Center project on Advanced Chatbot.

## 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.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. [Neural machine translation by jointly learning to align and translate](#). In proceedings of ICLR 2015 as oral presentation.
- Michael Bloodgood and Chris Callison-Burch. 2010. [Bucking the trend: Large-scale cost-focused active learning for statistical machine translation](#). In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, page 854–864, USA. Association for Computational Linguistics.

- 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 kučera 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.
- Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. [On the properties of neural machine translation: Encoder–decoder approaches](#). In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.
- 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.
- Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. [Active learning for statistical phrase-based machine translation](#). In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 415–423.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. [Long short-term memory](#). *Neural Comput.*, 9(8):1735–1780.
- Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. 2011. [Bayesian active learning for classification and preference learning](#). *CoRR*, abs/1112.5745.
- W. John Hutchins. 2004. [The georgetown-ibm experiment demonstrated in january 1954](#). In *Machine Translation: From Real Users to Research*, pages 102–114, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Philipp Koehn. 2005. [Europarl: A Parallel Corpus for Statistical Machine Translation](#). In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.
- 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, Claudia Leacock, and Christiane Fellbaum. 1998. [Building semantic concordances. WordNet: An Electronic Lexical Database](#), pages 199–216.
- David D. Lewis and Jason Catlett. 1994. [Heterogeneous uncertainty sampling for supervised learning](#). In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156, San Francisco (CA). Morgan Kaufmann.
- Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. [Learning to actively learn neural machine translation](#). In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 334–344, Brussels, Belgium. Association for Computational Linguistics.

- 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.
- Laurent Nepveu, Guy Lapalme, Philippe Langlais, and George Foster. 2004. [Adaptive language and translation models for interactive machine translation](#). In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 190–197, Barcelona, Spain. Association for Computational Linguistics.
- 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.
- Daniel Ortiz-Martínez. 2016. [Online learning for statistical machine translation](#). *Computational Linguistics*, 42(1):121–161.
- 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.
- Álvaro Peris and Francisco Casacuberta. 2018. [Active learning for interactive neural machine translation of data streams](#). In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 151–160, Brussels, Belgium. Association for Computational Linguistics.
- Matthew S. Ryan and Graham R. Nudd. 1993. [The viterbi algorithm](#). Technical report, University of Warwick.
- 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.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. [Sequence to sequence learning with neural networks](#). In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14*, page 3104–3112, Cambridge, MA, USA. MIT Press.
- 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 406–414, 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*.
- Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. [Modeling coverage for neural machine translation](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

# Active<sup>2</sup> Learning: Actively reducing redundancies in Active Learning methods for Sequence Tagging and Machine Translation: Appendix

## A Additional Remarks

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

### A.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, 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.

### A.2 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.

## B Hyper-parameters and other Implementation Details

Similar hyper-parameter values work across all the tasks. Hence, the same values were used for all experiments and these values were determined

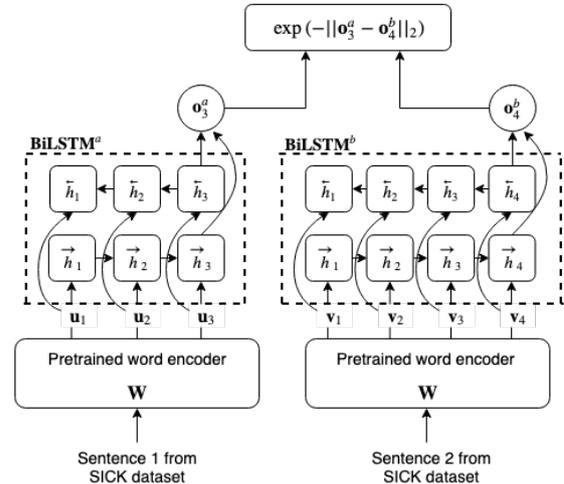


Figure 6: Modeling similarity using the Siamese encoder (enclosed by dotted lines). A pair of sentences from SICK dataset is fed to the pretrained sequence tagging model. The output of the word encoder is then passed to the Siamese encoder. Last hidden state of the Siamese encoder, corresponding to the sequence length of the sentence, is used for assigning a similarity score to the pair.

using the validation set of the 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 results. We use glove embeddings (Pennington et al., 2014) for all datasets. We apply normal dropout in the character encoder instead of the use of recurrent dropout (Gal and Ghahramani, 2016b) in the word encoder of the 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 the AL strategy. We chose two representative examples from each cluster.

<b>Active Learning strategy</b>	
threshold (Margin)	15
threshold (Entropy)	40
threshold (BALD)	0.2
dropout (BALD)	0.5
number of forward passes (BALD)	51
<b>Sequence tagging model</b>	
CNN filter sizes	[2,3]
training batch size	12
number of train epochs	16
dimension of character embedding	100
learning rate (Adam)	0.005
learning rate decay	0.9
<b>Siamese encoder</b>	
training batch size	48
number of train epochs	41
train/dev split	0.8
learning rate (Adam)	1e-5
period (of retrain)	10
<b>Clustering</b>	
Number of clusters	20
<b>Training</b>	
Batch size	12
<b>NMT model</b>	
training batch size	128
number of train epochs	20
dimension of (sub)word embedding	256
learning rate (Adam)	1e-3
<b>Siamese encoder</b>	
training batch size	1150
number of train epochs	25
dimension of (sub)word embedding	300
learning rate (Adam)	1e-3
period (of retrain)	3
<b>Clustering</b>	
Number of clusters	50
<b>Training</b>	
Batch size	128

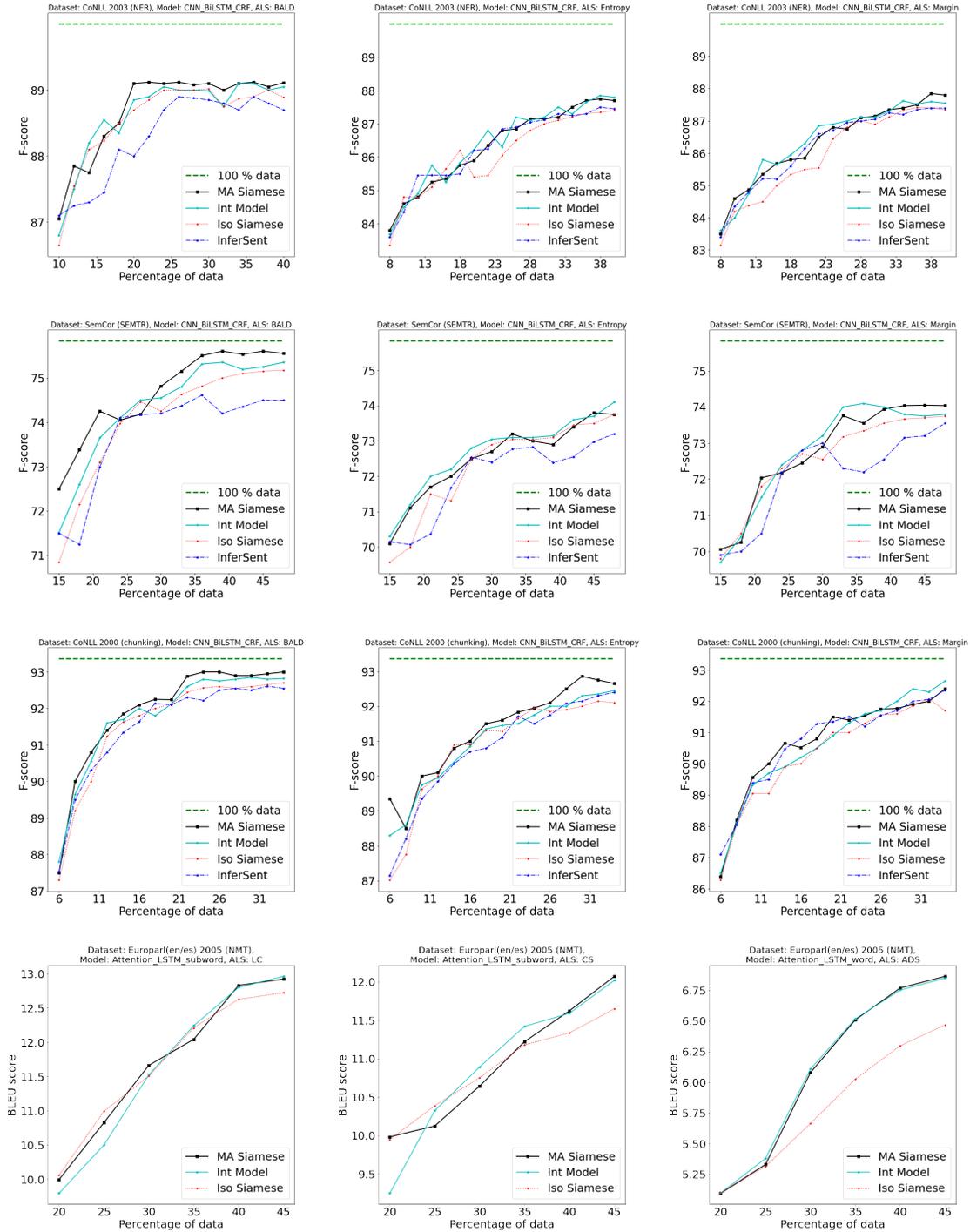


Figure 7: [Best viewed in color] Ablations studies on different tasks using different active learning strategies. 1<sup>st</sup> row: NER, 2<sup>nd</sup> row: SEMTR, 3<sup>rd</sup> row: CHUNCK, 4<sup>th</sup> row: NMT. In first three rows, from left to right, the three columns represent BALD, Entropy and Margin AL strategies. 4<sup>th</sup> row represents AL strategies for NMT, from left to right (LC: Least Confidence, CS: Coverage Sampling, ADS: Attention Distraction Sampling). Legend Description {100% data : full data performance, A<sup>2</sup>L (MA Siamese) : Model Aware Siamese, A<sup>2</sup>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.

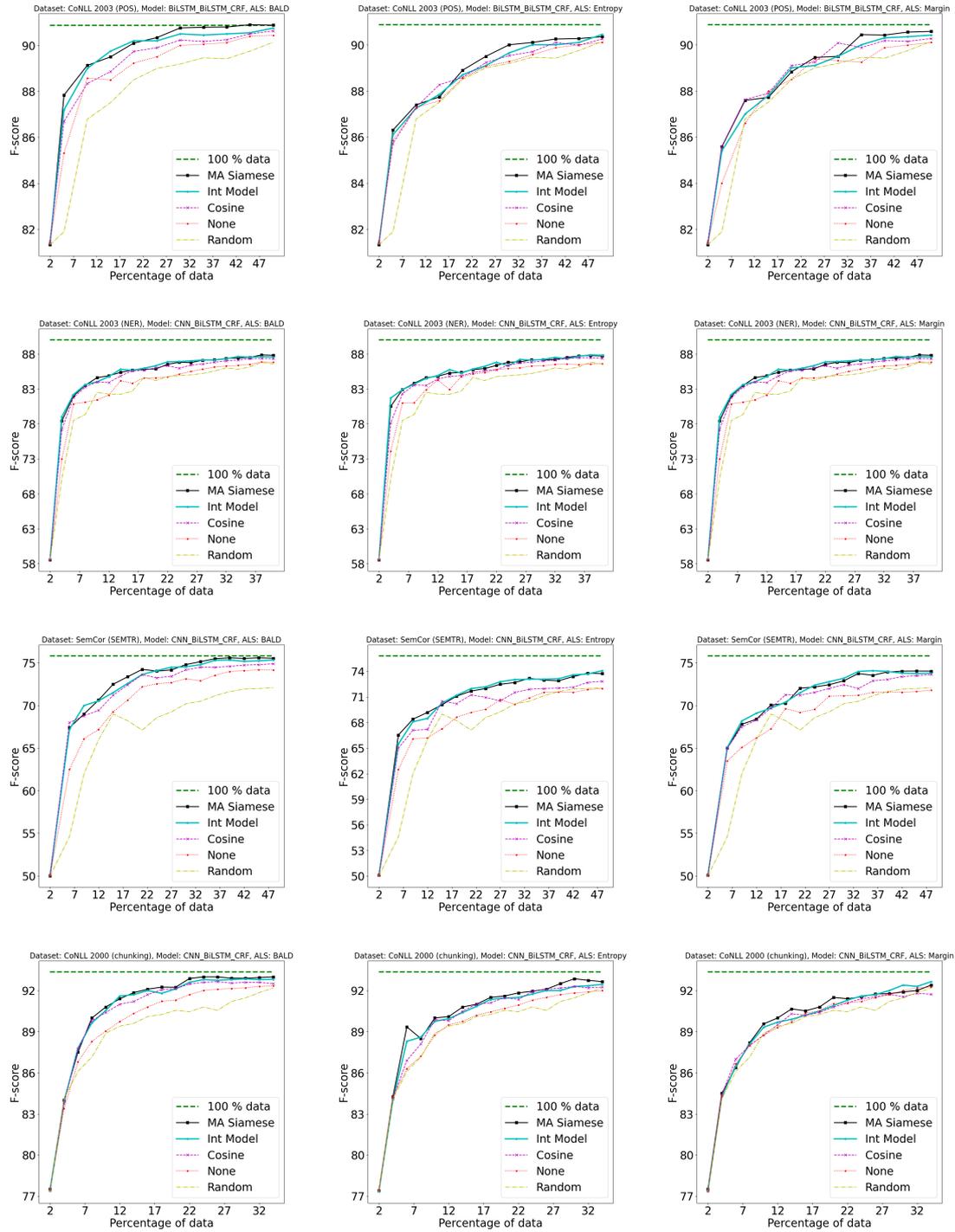


Figure 8: [Best viewed in color] Comparison of our approach (A<sup>2</sup>L) with baseline approaches on different tasks using different active learning strategies. 1<sup>st</sup> row: POS, 2<sup>nd</sup> row: NER, 3<sup>rd</sup> row: SEMTR, 4<sup>th</sup> row: CHUN. In each row, from left to right, the three columns represent BALD, Entropy and Margin based AL strategies. Legend Description {100% data : full data performance, A<sup>2</sup>L (MA Siamese) : Model Aware Siamese, A<sup>2</sup>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.

<b>Setup</b> \ <b>% data</b>	<b>10%</b>	<b>15%</b>	<b>20%</b>	<b>25%</b>	<b>30%</b>	<b>35%</b>	<b>40%</b>	<b>45%</b>	<b>50%</b>
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
<b>A<sup>2</sup>L (MA Siamese)</b>	<b>89.13</b>	89.50	90.10	<b>90.34</b>	<b>90.76</b>	<b>90.79</b>	<b>90.80</b>	90.70	<b>90.88</b>
<b>A<sup>2</sup>L (Int Model)</b>	89.00	<b>89.75</b>	<b>90.20</b>	90.20	90.50	90.45	90.50	<b>90.75</b>	90.75

Table 4: Interpretation of the plot on the top left corner of Fig 8 (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<sup>2</sup>L (MA Siamese) consistently performs better than other baselines.