Minimizing Annotation Effort via Max-Volume Spectral Sampling

Ariadna Quattoni
Universitat Politècnica de Catalunya
Barcelona, Spain
aquattoni@cs.upc.edu

Xavier Carreras
IIIA-CSIC
Barcelona, Spain
xavierc@iiia.csic.es

Abstract

We address the annotation data bottleneck for sequence classification. Specifically we ask the question: if one has a budget of N annotations, which samples should we select for annotation? The solution we propose looks for diversity in the selected sample, by maximizing the amount of information that is useful for the learning algorithm, or equivalently by minimizing the redundancy of samples in the selection. This is formulated in the context of spectral learning of recurrent functions for sequence classification. Our method represents unlabeled data in the form of a Hankel matrix, and uses the notion of spectral max-volume to find a compact sub-block from which annotation samples are drawn. Experiments on sequence classification confirm that our spectral sampling strategy is in fact efficient and yields good models.

1 Introduction

In the later years the field of NLP has witnessed great progress on supervised machine learning methods for sequence classification. However, most of these methods require large amounts of annotated training data. Because of this, whenever a new NLP application needs to be developed, data annotation becomes the main bottleneck in terms of cost and time. For example, a defense research analyst might wish to quickly train a text classifier to detect emergent socio-political events in a given conflict area. Since there might be only a few experts on the subject their time will be costly. Therefore, the expert should be able to train models fast with minimal annotation effort.

To address the annotated data bottleneck, researchers have proposed active learning approaches that develop sampling strategies designed to minimize the number of annotations required to train a model (Settles, 2009; Wang and Shang, 2014; Zhang et al., 2016; Siddhant and Lipton, 2018). Most active learning proposals are based on two main strategies. The first strategy uses model uncertainty and selects samples for which the prediction of the current model is the least confident. This strategy might not work very well during the first iterations of active learning, when the predictions of the model are unstable. Furthermore, the model uncertainty criteria cannot be applied in the first iteration, when no model has been trained and one needs to resort to other *cold start* sampling strategies (Yuan et al., 2020). To overcome the limitations of the uncertainty approach other researchers have proposed sampling strategies that attempt to maximize the diversity of the selected samples (Shao et al., 2019).

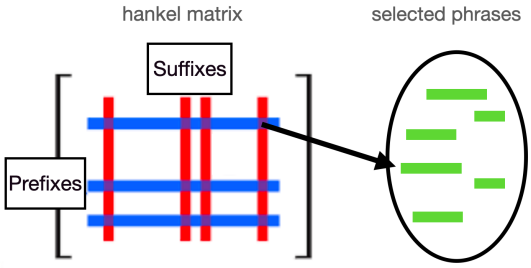


Figure 1: Representing unlabeled data in the form of a Hankel matrix can be very effective to uncover latent structure of the data. We present a sampling technique to leverage this structure.

Besides the selection strategy, another dimension of an active learning method is the type of annotation feedback that it exploits. For example, in text classification the annotations can consist of labels for complete texts, phrases, sentences, features, rules or labeling functions (McCallum and Nigam, 1999; Settles et al., 2008; Druck et al., 2009; Ratner et al., 2017; Safranchik et al., 2020).

In this paper we focus on the problem of training sequence classification models under an annotation

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2890–2899
November 7–11, 2021.

budget constraint and with no prior trained model for the task. This is sometimes referred as the *cold start problem*. We consider a setting in which the annotation feedback is at the level of phrases. Our goal is to develop an efficient algorithm that can answer the question: if one has a budget of N annotations, which phrases should we select to annotate?

Notice that active sampling under an annotation budget is different from the classical active learning scenario. In the classical setting, the learning algorithm has access to a large unlabeled data pool, and in a series of iterations it alternates between sampling data to annotate and training a new model. In contrast, when training under budget constraints, the focus is on the initial setting, when there is no model that can guide the selection and when all that is available to the selection algorithm is the unlabeled data pool. The second difference is that our goal is to find the optimal set of size N, i.e. the selection criteria should be able to score a set or batch of phrases. In that sense our work is more related to cover-set approaches (Sener and Savarese, 2017).

Our proposed solution for the problem of learning under a budget constraint follows a diversity sampling strategy. That is, given a fixed budget our batch selection method attempts to maximize the amount of useful information contained in the batch. Or equivalently, it tries to minimize annotation redundancy in the selected batch. More precisely, our approach is inspired in methods that minimize annotation redundancy by uncovering latent structure in the input domain (Dasgupta and Hsu, 2008).

Intuitively, imagine a classification problem with k classes. If we had access to a clustering of the data into k groups that perfectly align with the target classes only k labeled points would be needed. That is, we would label a representative sample for each cluster. Of course, the perfect clustering might not exist but the point is that by discovering relevant latent structure one can minimize annotation redundancy and ask only for the annotations that are really necessary.

We take this basic idea and translate it to the sequence classification setting. Essentially, our method induces an implicit soft clustering of phrases (i.e., subsequences) so that we only need annotations for one phrase in each cluster. Following the classical distributional principle, we consider two phrases to be similar if they can appear in similar contexts.

Our technical contribution exploits ideas from the theory of spectral and Hankel-based learning methods for estimating recurrent sequence prediction functions with linear state-dynamics (Hsu et al., 2009; Bailly et al., 2009; Balle et al., 2014; Rabusseau et al., 2019). We reduce the problem of training sequence classification models under annotation budget constraints to the problem of selecting a high-volume matrix sub-block (i.e. a sub-block of high rank) from a Hankel matrix (computed from the unlabeled pool) that captures key statistics of the sequence domain distribution. See Figure 1 for a sketch. To our knowledge, sampling strategies reduced to spectral matrix operations is a novel technical approach. Recent methods for cold-start sampling with an annotation budget have considered clustering embeddings of sentences derived from BERT (Yuan et al., 2020), either as a single shot sampling (like our method), or by iterative fine-tuning of the embeddings used for sampling.

We highlight two main contributions:

- We propose a notion of *sample diversity* based on structural properties of low-rank Hankel matrices. Using this notion we derive a phrase-sampling algorithm for learning under annotation budget constraints, i.e. the cold-start challenge.

- In experiments, we compare our spectral sampling strategy to recent active learning methods for fine-tuning BERT-based sentence classifiers, that also seek diversity in the sampling step. Our results show that under strict budget constraints a simple latent-state model (in our case, linear RNNs) can outperform the neural BERT-based approach, despite the fact that our models are strictly less expressive and are not pre-trained.

The paper is organized as follows: Section 2 starts with a description of linear recurrent sequence functions, and then provides the spectral learning background necessary to understand our sampling strategy, and in particular the concept of Hankel matrices. Section 3 presents the proposed phrase selection method based on selecting a max-volume sub-block from a Hankel matrix representing the unlabeled pool. Section 4 presents our experiments on text classification. Finally, Section 5 concludes the paper.

2 Linear RNNs and Hankel Matrices

In this paper we work with simple Recurrent Neural Networks that use linear functions (matrix products) to compute hidden-state vectors along the sequence. In this setting, we describe connections to spectral learning, and specifically to the Hankel matrix of a recurrent sequence model. This is a central tool to derive the sampling strategy.

A Recurrent Neural Network (RNN) takes as input a sequence x and outputs a vector of k real numbers, $f : \Sigma^\star \to \mathbb{R}^k$, where $x = x_1 \cdots x_n$ is a sequence of length n over some finite alphabet Σ. We denote as $\Sigma^\star$ the set of all finite sequences, and we use it as a domain of our functions. An RNN with hidden dimension d and output dimension k is defined as a tuple:

$$N = \langle \mathbf{h}_0, \mathbf{W}_\mathrm{h}, \mathbf{U}_\mathrm{h}, \mathbf{b}_\mathrm{h}, \mathbf{W}_\mathrm{y}, \mathbf{b}_\mathrm{y}, \gamma_\mathrm{h}, \gamma_\mathrm{y} \rangle \quad . \quad (1)$$

The parameters $\mathbf{h}_0 \in \mathbb{R}^d$, $\mathbf{W}_\mathrm{h} \in \mathbb{R}^{d \times |\Sigma|}$, $\mathbf{U}_\mathrm{h} \in \mathbb{R}^{d \times d}$, $\mathbf{b}_\mathrm{h} \in \mathbb{R}^d$ compute hidden vectors at each position t of the of the sequence

$$\mathbf{h}_t = \gamma_\mathrm{h}(\mathbf{W}_\mathrm{h}\mathbf{e}_{x_t} + \mathbf{U}_\mathrm{h}\mathbf{h}_{t-1} + \mathbf{b}_\mathrm{h}) \quad , \quad (2)$$

where $\mathbf{e}_\sigma$ is an indicator vector of the current symbol $\sigma \in \Sigma$ that selects the appropriate column weights in $\mathbf{W}_\mathrm{h}$. The parameters $\mathbf{W}_\mathrm{y} \in \mathbb{R}^{k \times d}$, $\mathbf{b}_\mathrm{y} \in \mathbb{R}^k$ model the k function values given the hidden vectors:

$$f_N(x_{1:n}) = \gamma_\mathrm{y}(\mathbf{W}_\mathrm{y}\mathbf{h}_n + \mathbf{b}_\mathrm{y}) \quad . \quad (3)$$

The functions γ_h and γ_y are activation functions, and in this paper we focus on simply using the identities.

The most common use of general RNNs in NLP is language modeling. In this case the model is set to predict the conditional probability of the next symbol (with $k = |\Sigma|$ symbols), and by means of the chain rule, the model defines a distribution over the language and is trained to generate sequences left-to-right. Another popular use is sequence classification, where the model computes a classification score for each of the k labels of a task, given input sequences $x_{1:n}$.

Another application of RNNs, which is less common in the literature, is to frame language modeling as a density estimation task, where the model estimates the probability of a full sentence directly. In this case, the RNN predicts a single score (i.e. $k = 1$) which corresponds to the probability of the input $x_{1:n}$, and we can regard this as a regression learning problem, i.e. learn a real-valued function that approximates the target probabilities given the hidden state vector of the input sequence.

Finally, instead of modeling the probability of full sequences, RNNs can directly approximate the moments of the distribution. That is, learn a function from $\Sigma^* \to \mathbb{R}$ that estimates the expected number of times of observing a subsequence $x_{1:n}$ in a random sequence sampled from the target distribution. Modeling moments, such as ngram statistics, has the advantage that the target statistics are less sparse than full sequences even for long ngrams.

2.1 Linear RNNs and Hankel Matrices

We now focus on linear RNNs where the activation functions γ_h and γ_y are simply the identity function.[1] We describe some interesting properties of linear RNNs that we exploit in the context of sampling.

A linear RNN N can be rewritten into a Weighted Finite Automata (WFA) of this form:

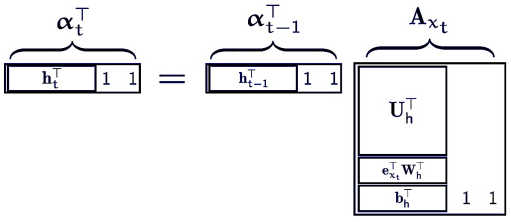
$$f_N(x_{1:n}) = \boldsymbol{\alpha}_0^\top \mathbf{A}_{x_1} \mathbf{A}_{x_2} \cdots \mathbf{A}_{x_n} \mathbf{B} \quad . \quad (4)$$

where: $\boldsymbol{\alpha}_0 \in \mathbb{R}^d$ is an initial state vector; $\mathbf{A}_\sigma \in \mathbb{R}^{d \times d}$ are the transition matrices associated with each symbol $\sigma \in \Sigma$; and $\mathbf{B} \in \mathbb{R}^{d \times k}$ is a matrix of state-to-output weights. One can verify that a linear RNN N with d' hidden states and k outputs can be rewritten as a WFA $\langle \boldsymbol{\alpha}_0, \mathbf{A}_\sigma, \mathbf{B} \rangle$ of dimension $d = d' + 2$ and k outputs.[2]

Note that under Eq. 4 the computation of a linear RNN is not necessarily in a forward manner (left-to-right), but can also be in a backward manner (right-to-left). Given an input sequence $x_{1:n}$, one can define *forward vectors* for prefixes of the sequence $\boldsymbol{\alpha}_t^\top = \boldsymbol{\alpha}_{t-1}^\top \mathbf{A}_{x_t}$; and *backward matrices* for suffixes of the sequence $\boldsymbol{\beta}_t = \mathbf{A}_{x_t} \boldsymbol{\beta}_{t+1}$. Then, for any position $1 \leq t \leq n$ we have that $f(x_{1:n}) = \boldsymbol{\alpha}_t^\top \boldsymbol{\beta}_{t+1}$.

[1] Our description should generalize to any linear activation function, but for simplicity we just use the identity function.

[2] The construction works by packing the $\mathbf{W}_\mathrm{h}$, $\mathbf{U}_\mathrm{h}$ and $\mathbf{b}_\mathrm{h}$ parameters into matrices $\mathbf{A}_\sigma$ for each $\sigma \in \Sigma$, using two dummy dimensions in the state vectors to carry the symbol and bias weights, as illustrated here:



Output parameters $\mathbf{W}_\mathrm{y}$ and $\mathbf{b}_\mathrm{y}$ are packed into $\mathbf{B}$ similarly.

We now focus on linear RNNs that compute a single output value, i.e. $k = 1$. We can represent a linear RNN using a Hankel matrix. A Hankel matrix of a sequence prediction function f is a bi-infinite matrix $\mathbf{H}_f \in \mathbb{R}^{\Sigma^\star \times \Sigma^\star}$ indexed by prefixes and suffixes of the language, such that $\mathbf{H}_f(p, s) = f(p \cdot s)$. A central result establishes that for a WFA that computes function f, with d dimensions and $k = 1$, the rank of $\mathbf{H}_f$ is d. This is because WFAs and linear RNNs factor the computation of f as products of prefix and suffix vectors, which are of dimension d. The reverse also holds: if a Hankel matrix $\mathbf{H}_f$ has rank d, then there is a WFA with d states that computes the associated f function. Next we describe spectral learning, which uses this result. See (Rabusseau et al., 2019) for further connections between WFAs and linear RNNs. See (Quattoni and Carreras, 2020) for an application of WFAs to NLP sentence classification tasks.

2.2 The Spectral Method

Spectral learning is based on learning a low-rank Hankel matrix of the target distribution. Here we provide a high level description of the method; for a complete derivation and the theory justifying the algorithm we refer the reader to the works by Hsu et al. (2009) and Balle et al. (2014).

At training, we are given sequences T from the distribution and we want to estimate f. We denote as $f_T(x)$ the empirical subsequence expectation of x in T.[3] Using f_T, the spectral method estimates a WFA A with d states, where d is a parameter of the algorithm, such that f_A is a good approximation of f. The method reduces the learning problem to computing an SVD decomposition of the *training Hankel matrix*, that collects the observed expectations f_T. The method is as follows:

(1) Select a set of prefixes P and of suffixes S, that serve as indices of the Hankel matrix for rows and columns respectively. For example, select all subsequences up to a certain size n.

(2) Create a Hankel matrix $\mathbf{H} \in \mathbb{R}^{P \times S}$ for the basis (P, S). Each entry is indexed by a prefix $p \in P$ and a suffix $s \in S$, and the value is the evaluation of f_T on the concatenation of the prefix and the suffix i.e. $\mathbf{H}(p, s) = f_T(p \cdot s)$.

(3) Compute a d-rank factorization of $\mathbf{H}$. Compute the SVD of $\mathbf{H}$, i.e. $\mathbf{H} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ resulting in a matrix $\mathbf{P} = \mathbf{U}\mathbf{\Sigma} \in \mathbb{R}^{P \times d}$ and a matrix $\mathbf{S} = \mathbf{V} \in \mathbb{R}^{S \times d}$. $\mathbf{H} \approx \mathbf{P}\mathbf{S}^\top$ is a d-rank factorization of $\mathbf{H}$, with $\mathbf{P}$ and $\mathbf{S}$ being projection matrices of prefixes and suffixes (respectively) to d-dimensional embeddings.

(4) Recover the WFA A of d states using the previous Hankel factors $\mathbf{P}$ and $\mathbf{S}$ (details omitted).

The steps above are only a sketch of the method, a full description can be found in (Balle et al., 2014). The main computation is dominated by step (3), the SVD of the Hankel matrix, which is at most cubic in the size of the matrix.

One could imagine a Hankel matrix of infinite size, which would capture the statistics of all of the training subsequences. The theory behind spectral learning shows that this infinite Hankel matrix, when representing a function computable by a minimal WFA of d states, has rank d. Furthermore, the theory shows that *any* sub-block of the infinite Hankel that preserves the rank (i.e. that has rank d) is sufficient to learn the target WFA. This observation sheds light on step (1) of the algorithm above: it attempts to define a finite sub-block of the infinite Hankel (by defining a finite basis of prefixes S and suffixes P) that preserves its rank.

In theory, we can define a Hankel matrix that captures all of the data by setting both P and S to be all subsequences found in *any* training sequence. However, this results in a very large Hankel matrix, which has a consequence on the cost of the SVD in step (3). There exist techniques to handle this computational bottleneck (Quattoni et al., 2017).

3 Phrase Sampling via Max-Volume Optimization

In this section we describe a deterministic phrase sampling method for sequence classification. We assume an unlabeled pool of sequences, and the goal is to select an annotation batch. Once selected, the batch will be first annotated, and then a model will be learned from it. The sampling strategy we describe selects phrases for annotation, i.e. subsequences of sentences (i.e. ngrams) found in the unlabeled pool.

Notice that when learning a sequence model with the spectral method, we only use the information contained in the selected Hankel sub-block. In the previous section we discussed the importance of

[3] This corresponds to the number of times that x is observed as subsequence of any sequence in T, normalized by the number of sequences in T.

selecting a small sub-block of the Hankel matrix for computational efficiency. In such setting, it is assumed that there is enough training data to compute all the entries of the Hankel. A sub-block of $\mathbf{H}$ is selected to ease the computation of the SVD, which is required to infer the model parameters.

The problem that we address in this paper is different: the focus is annotation efficiency, not computational efficiency. In our case, we assume that we need to estimate the Hankel matrices $\mathbf{H}_l$ of each label l. Initially, we do not have samples to compute any of its entries, so we ask the question: Is there a way to select the samples to annotate so that it provides the most information about the target class distributions? Or equivalently, is there a way to request annotations so that it gives us the most information about $\mathbf{H}_l$?

Our solution uses an approximation of $\mathbf{H}_l$ given by the Hankel matrix $\mathbf{H}_U$ associated with a language model of the unlabeled distribution. We use $\mathbf{H}_U$ to pick the most informative entries of $\mathbf{H}_l$, i.e. those for which we will request annotations. Intuitively, each prefix in the matrix is described as a distribution over suffixes, and the analogue for suffixes. The proposed approach will select prefix and suffix rows that are the most uncorrelated, so that annotating their compositions will provide the most information about $\mathbf{H}_l$. In essence, this selects a set of representative prefix and suffix prototypes in the latent space of prefix and suffix embeddings derived from $\mathbf{H}_U$.

The difference between the computational and sampling problems has an analogue in recommendation systems based on collaborative filtering. In this case, one creates a matrix where rows are users and columns are movies, and the corresponding entry has the rating given by a user to a movie. Some entries are observed and some are missing, the matrix is assumed to be low-rank, and the goal is to complete the matrix and predict the ratings that users will give to unseen movies. In this context, the computational challenge is to perform SVD of a potentially very large matrix, which is required for low-rank matrix completion. In contrast, the sampling problem, assumes that we can query users for ratings on specific movies. The optimal sampling question is: What is the most informative subset of user-movie ratings to request so that from the chosen subset we can predict unseen user-movie ratings?

3.1 The Max-Volume Hankel Sub-block

First, we are interested in having an annotation budget. This budget could be defined in terms of the number of tokens to annotate. However, because of reasons that will become apparent, in our spectral approach it is more natural to define a budget on the size of the sub-block; the number of tokens to annotate will be determined by it.

We redefine the spectral algorithm to work with Hankel sub-blocks of size $b \times b$, where b is the budget. Given a large Hankel matrix, it is known that finding the sub-block of size b of maximum rank is NP hard (Peeters, 2003). Fortunately there exist reasonable approximations. A popular approach that is often used in the context of reconstruction of low-rank matrices under computational constraints is to search for the sub-block of highest volume, where the matrix volume is defined as the absolute value of the determinant, i.e. the product of the Eigenvalues (Bebendorf, 2000; Çivril and Magdon-Ismail, 2009; Cortinovis et al., 2019). While finding a sub-block of maximum volume is also NP-hard, there exist efficient and widely used approximation algorithms. In this paper we use an iterative algorithm based on LU factorization (Miranian and Gu, 2003). It iteratively factors matrices of size $n \times b$, where n is the number of rows/columns of the original Hankel matrix and b is the budget. In our experiments, this routine takes a few minutes to converge.

3.2 Max-Volume Sub-block for Sampling

We now turn to using max-volume as a sampling strategy for sequence classification, under an annotation budget. The classifiers we use are ensembles of linear RNNs, with one model for each label trained to estimate the class-specific moments. We could attempt to select a Hankel sub-block for each label, but the sub-block selection methods we described would require a large Hankel matrix specific to each label, which in turn would require large labeled training data.

The main idea behind our sampling strategy is to have a single sub-block that is common to all the labels, and to make this selection we use the distribution of unlabeled sequences in the domain. Specifically, we first consider a Hankel matrix $\mathbf{H}_U$ of the sequences in the unlabeled pool, where the value of an entry $\mathbf{H}_U(p, s)$ is the expected number of times of observing the phrase $p \cdot s$ in a sequence sampled from the unlabeled pool. This corresponds

to a Hankel matrix for language modeling, since it is estimating the domain distribution. This Hankel matrix is used to select a max-volume sub-block satisfying the given budget b, resulting in a set of b prefixes P and a set of b suffixes S. This basis is then used to define a Hankel matrix specific to each label.

3.3 Filling in Hankel Matrices

For each label l, we need to fill all the entries of the associated Hankel matrix $\mathbf{H}_l$, defined over the max-volume basis. The value of one entry $\mathbf{H}_l(p, s)$ corresponds to the expected number of times of observing the phrase $p \cdot s$ in a sequence sampled from the specific distribution of all sequences of label l. For each possible phrase defined by the basis, and for each label l, we would need such statistic. It seems unrealistic to ask an annotator this kind of feedback.

Instead, we note that because of the Zipfian nature of language, rather than requiring the actual expectation of a phrase, in many cases it suffices to know whether that expectation is 0 or not, i.e. whether or not that phrase can appear in sequences of that class. Put it differently, we postulate that most of the information is in the sparsity pattern of the moments in the Hankel matrix, rather than their real values.

Designing an annotation strategy to fill sparsity patterns is much easier. For each prefix $p \in P$ and suffix $s \in S$ we consider the phrase $q = p \cdot s$. If q does not appear in the unlabeled pool we set $\mathbf{H}_l(p, s) = 0$ for all labels l. Otherwise, if q does appear in the unlabeled pool we ask the annotator for its class labels. We use multilabel-style feedback where we allow a phrase q to belong to multiple classes simultaneously, and set $\mathbf{H}_l(p, s) = 1$ to all such positive labels l, and 0 for the rest of labels. Algorithm 1 describes the sampling strategy.

We would like to note that once we have selected an informative basis for the sequence classification task at hand, other annotation feedback strategies might be used to fill the necessary Hankel statistics, for example by generating phrases. In this work we picked the simplest strategy from which we obtained good performance, further work will explore other strategies.

Algorithm 1: Phrase Sampling via Max-Volume Optimization

Data: Unlabeled data pool U, basis budget b, a set L of k target labels

1 Compute Hankel matrix $\mathbf{H}_U$ where rows and columns are indexed prefixes and suffixes
2 Find maximum-volume matrix sub-block of $\mathbf{H}_U$ and corresponding basis (P, S) where $|P| = |S| = b$
3 Construct the set of queries Q by listing all phrases $p \times s$ obtained by concatenating a prefix $p \in P$ with a suffix $s \in S$, such that $p \times s$ is observed in U
4 For every phrase $q \in Q$ ask the annotator to provide feedback, in the form of an indicator vector $z \in \{0, 1\}^k$, where z_l denotes that q can be a phrase of sentences of class $l \in L$

Result: A set of labeled phrases $\{(q, z) \mid q \in Q, z \in \{0, 1\}^k\}$

4 Experiments

We evaluate the spectral sampling method on two sentence classification tasks. We compare our sampling strategy to recent active learning methods for fine-tuning BERT-based sentence classifiers (Yuan et al., 2020), that also seek diversity in the sample.

Data. We use two common datasets for sentence classification: the IMDB dataset of movie reviews (Maas et al., 2011), where the goal is to predict if a movie review is positive or negative; and the AG News dataset Zhang et al. (2015) of news articles headlines classified into four classes. The IMDB dataset has $17,500$ training examples, $7,500$ validation examples, and $25,000$ test examples. The AG News dataset has $110,000$ training examples, $10,000$ validation examples and $7,600$ test examples. We use the union of the training and validation as the unlabeled pool of examples.

Evaluation. We report performance on the test partition. As an evaluation metric we use the F1 average between precision and recall. We report the F1 performance of the model as a function of the total number of tokens annotated, defined as $\sum_{q \in Q} |q|$ where Q is the set of annotated samples. Since our sampling method is controlled by a budget on the basis size, we run the method for increasing budgets and measure the number of tokens of each batch of samples.

Linear RNN Classifiers. We trained one linear RNN for each class, that models the distribution of sequences of that class. To train them, we use the spectral method of moments, and set the number of

hidden states to 10 for each of them. We could, in principle, exploit models with larger state spaces. If large quantities of data were available we would have indeed observed a performance improvement by exploiting larger state spaces. However, we decided to use a small state space since our main goal is to be able to train models with small annotation budgets. Under this setting simpler models can be learned more robustly. Given an input sequence x, the linear RNN classifiers provide scores for each ngram (i.e. substring) of x and each class. To make predictions, we use a simple ensemble technique similar to (Mesnil et al., 2014): we consider all the ngrams of x up to length 4, and compute an aggregate prediction score for each label $l \in L$:

$$z(x, l) = \sum_{w \in \text{ngrams}(x)} \frac{f_l(w)}{\sum_{l' \in L} f_{l'}(w)} \quad . \quad (5)$$

Simulated Annotation. Our sampling method produces a batch of phrases (i.e. subsequences of unlabeled examples) for annotation. While doing evaluations with human annotators would be ideal, it is also very costly. Instead we follow the standard evaluations of active learning methods which are based on using the unlabeled pool together with the true labels to simulate the feedback that could be provided by a human annotator. While this is by no means perfect, it is a natural low-cost approximation to the human evaluation experiment. More precisely, for a given phrase q to be annotated we look at the unlabeled data pool and retrieve the sentences in which q appears. Then we take all the labels for such sentences and set them as positive labels for q, forming an indicator vector.

4.1 Comparison to Max-Volume Oracles

We first test the max-volume sampling using oracle configurations that have access to fully labeled data. The oracle max-volume is as follows. Since we have fully labeled data, we can consider class-specific Hankel matrices for each label. Thus, for each label we will compute the max-volume sub-block. We call this the *class-oracle* setting. Then, based on the discussion in Section 3.3, we consider two variants depending on how we fill the selected sub-blocks with target values. In *class-oracle expectations* the values are the expected counts of the corresponding phrases in the unlabeled pool. In *class-oracle occurrences* we only consider the sparsity pattern, setting 1 if the expected count is non-zero, and 0 otherwise.

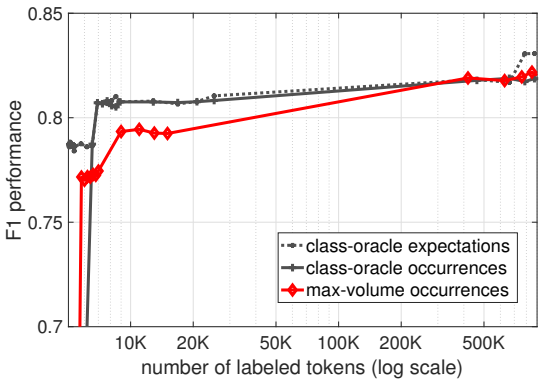


Figure 2: Comparison to max-volume oracles for phrase sampling on IMDB data.

Figure 2 shows F1 performance in terms of the number of annotated tokens. We clearly see that using occurrences behaves very similarly to using actual expectations. This confirms our hypothesis, and enables to train our models from simple binary phrase occurrence feedback. The same figure also shows the curve for our proposed sampling method, that estimates the max-volume sub-block using unlabeled sequences. We can see it follows the same trend as the oracles. This confirms that using the underlying domain distribution to inform about sub-blocks of maximum information is indeed an effective working hypothesis.

4.2 Comparisons for Fixed Annotation Budgets

We now compare our strategy for sampling under budget constraints with two baselines. The first baseline samples complete *random examples*, and the second one samples *random phrases* of length less than 10.

We also compare to three active learning methods analyzed by Yuan et al. (2020) that also look for diversity in the queried samples: BERT-KM generates samples based on a k-means clustering of BERT embeddings of sentences, while BADGE and ALPS actively refine the BERT embeddings to the target task after getting labels for each batch of samples. The idea behind BADGE (Ash et al., 2020) is to use gradient representations of the sentences in the unlabeled pool, since gradients are indicators of changes in the model and therefore are useful to promote diversity. The ALPS method is a variant that uses the masked language modeling loss of BERT to promote gradient diversity for sampling purposes. In all, these methods represent

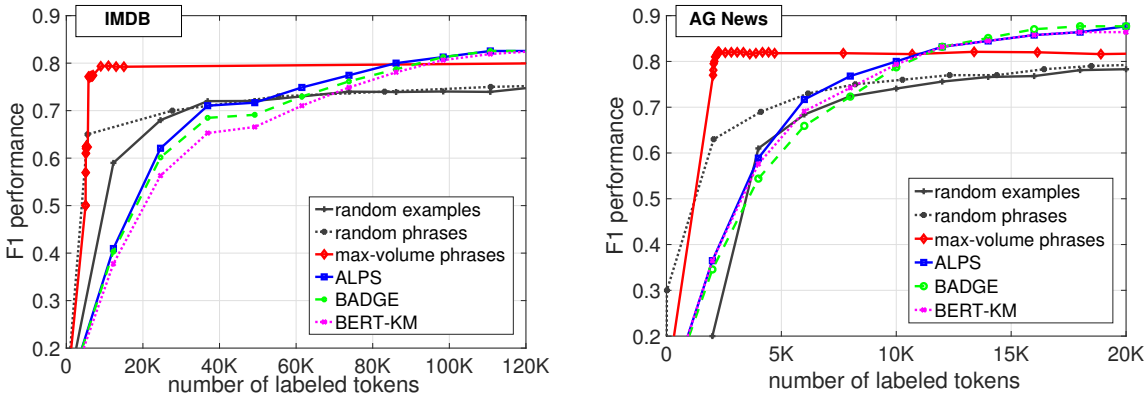


Figure 3: Comparison with alternative sampling methods under fixed annotation budget, on IMDB (left) and AG News (right) datasets.

basis size	#tokens	F1
11	5,056	50%
30	5,151	57%
50	5,251	61%
200	6,001	62%
400	7,001	77%
1,000	10,001	77%
2,000	15,001	79%
5,000	30,121	79%

Table 1: Performance of max-volume sampling in IMDB data with respect to the basis budget, and corresponding number of tokens for annotation.

recent BERT-based approaches for cold-start sampling. Our comparison follows the same setting as Yuan et al. (2020).

Figure 3 shows the comparison. The main observation is that max-volume sampling is much more efficient than the two baselines. Compared to the BERT-based samplers, we also see that max-volume sampling is more efficient for low budget settings, even though after some iterations, ALPS, BADGE and BERT-KM eventually outperform the accuracy of our method. One possible reason is that these methods do exploit information that is not in the max-volume sub-block. Table 1 shows in more detail the performance of max-volume sampling in terms of the size of the basis and the total number of tokens to be labeled.

5 Conclusions

Sequence distributions that can be modeled with latent state models have low-rank signatures. That is, the whole distribution can be learned from statistics over a small number of key phrases. The main contribution of our work is to show how we can leverage that property to design efficient sampling strategies for sequence classification under annotation budget constraints.

The idea is quite simple: while for a given category we cannot know a priori (that is without labeled sequences) its low-rank signature (and key phrases), we can try to estimate the signature from unlabeled domain data. Using that approximation we can design an efficient way of selecting phrases to label. Our experiments show that with this strategy we can obtain reasonable sequence classification models under small budget constraints. To the best of our knowledge our proposal is the first sampling strategy to implicitly exploit low-rank embeddings of domain phrases.

Once a low-rank Hankel signature has been found we could imagine several different annotation strategies for estimating the relevant statistics. This work is just a first step where we consider one of the simplest of such strategies. However, future work should explore the space of annotation strategies taking into account what feedback would result in the best estimation, and what is easiest for the human annotator. We see this work as opening the door for future research on interactive machine learning for sequence modeling where the annotation feedback strategy is designed to exploit the structural properties of the domain.

Our sampling strategy contrasts with recent work in active learning, which exploits BERT-based embeddings. We empirically observed that the performance of our combo is better for very low annotation budget, but eventually the neural approaches improve and gradually achieve state-of-the-art results. Thus, one natural question for future research is if our sampling strategy can be coupled with more expressive neural classifiers. A second re-

lated question is how to use the spectral models trained with max-volume sampling to warm-start neural approaches.

Acknowledgements

This work has been supported by the European Research Council under StG grant 853459 (INTERACT).

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*.

Raphaël Bailly, François Denis, and Liva Ralaivola. 2009. Grammatical inference as a principal component analysis problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 33–40, New York, NY, USA. ACM.

Borja Balle, Xavier Carreras, Franco M. Luque, and Ariadna Quattoni. 2014. Spectral Learning of Weighted Automata: A Forward-Backward Perspective. *Machine Learning*, 96(1):33–63.

Mario Bebendorf. 2000. Approximation of boundary element matrices. *Numerische Mathematik*, 86(4):565–589.

Alice Cortinovis, Daniel Kressner, and Stefano Massei. 2019. On maximum volume submatrices and cross approximation for symmetric semidefinite and diagonally dominant matrices. *arXiv e-prints*, page arXiv:1902.02283.

Sanjoy Dasgupta and Daniel Hsu. 2008. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215.

Gregory Druck, Burr Settles, and Andrew McCallum. 2009. Active learning by labeling features. In *Proceedings of the 2009 conference on Empirical methods in natural language processing*, pages 81–90.

Daniel J. Hsu, Sham M. Kakade, and Tong Zhang. 2009. A spectral algorithm for learning hidden markov models. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Andrew McCallum and Kamal Nigam. 1999. Text classification by bootstrapping with keywords, EM and shrinkage. In *Unsupervised Learning in Natural Language Processing*.

Grégoire Mesnil, Tomas Mikolov, Marc'Aurelio Ranzato, and Yoshua Bengio. 2014. Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews. *arXiv preprint arXiv:1412.5335*.

L Miranian and Ming Gu. 2003. Strong rank revealing lu factorizations. *Linear algebra and its applications*, 367:1–16.

René Peeters. 2003. The maximum edge biclique problem is np-complete. *Discrete Appl. Math.*, 131(3):651–654.

Ariadna Quattoni and Xavier Carreras. 2020. A comparison between CNNs and WFAs for sequence classification. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 159–163, Online. Association for Computational Linguistics.

Ariadna Quattoni, Xavier Carreras, and Matthias Gallé. 2017. A Maximum Matching Algorithm for Basis Selection in Spectral Learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1477–1485, Fort Lauderdale, FL, USA. PMLR.

Guillaume Rabusseau, Tianyu Li, and Doina Precup. 2019. Connecting weighted automata and recurrent neural networks through spectral learning. In *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1630–1639. PMLR.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

Esteban Safranchik, Shiying Luo, and Stephen Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5570–5578.

Ozan Sener and Silvio Savarese. 2017. Active Learning for Convolutional Neural Networks: A Core-Set Approach. *arXiv e-prints*, page arXiv:1708.00489.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles, Mark Craven, and Lewis Friedland. 2008. Active learning with real annotation costs. In *Proceedings of the NIPS workshop on cost-sensitive learning*, pages 1–10. Vancouver, CA:.

Jingyu Shao, Qing Wang, and Fangbing Liu. 2019. Learning to Sample: an Active Learning Framework. *arXiv e-prints*, page arXiv:1909.03585.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian Active Learning for Natural Language Processing: Results of a Large-Scale Empirical Study. *arXiv e-prints*, page arXiv:1808.05697.

D. Wang and Y. Shang. 2014. A new active labeling method for deep learning. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 112–119.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.

Ye Zhang, Matthew Lease, and Byron C Wallace. 2016. Active discriminative text representation learning. *arXiv preprint arXiv:1606.04212*.

Ali Çivril and Malik Magdon-Ismail. 2009. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theor. Comput. Sci.*, 410(47-49):4801–4811.