Deep Learning for Period Classification of Historical Hebrew Texts

Chaya Liebeskind and Shmuel Liebeskind

Jerusalem College of Technology, Lev Academic Center, Israel

Corresponding author: Chaya Liebeskind , liebchaya@gmail.com

Abstract

In this study, we address the interesting task of classifying historical texts by their assumed period of writing. This task is useful in digital humanity studies where many texts have unidentified publication dates. For years, the typical approach for temporal text classification was supervised using machine-learning algorithms. These algorithms require careful feature engineering and considerable domain expertise to design a feature extractor to transform the raw text into a feature vector from which the classifier could learn to classify any unseen valid input. Recently, deep learning has produced extremely promising results for various tasks in natural language processing (NLP). The primary advantage of deep learning is that human engineers did not design the feature layers, but the features were extrapolated from data with a general-purpose learning procedure. We investigated deep learning models for period classification of historical texts. We compared three common models: paragraph vectors, convolutional neural networks (CNN) and recurrent neural networks (RNN), and conventional machine-learning methods. We demonstrate that the CNN and RNN models outperformed the paragraph vector model and the conventional supervised machine-learning algorithms. In addition, we constructed word embeddings for each time period and analyzed semantic changes of word meanings over time.

Keywords

Machine Learning; Deep Learning; Period Classification; Diachronic Corpus

I INTRODUCTION

The aim of preserving and rendering cultural heritage more accessible motivates the digitization of historical texts in the last decade. Owing to the increasing availability of historical texts in digital form, the application of natural language processing (NLP) methods and tools to historical texts have garnered particular interest. Particular linguistic properties of historical texts, including nonstandard grammar, orthography, and abbreviations often pose challenges for NLP. Another challenge is predicting the period when a text was originally written. We herein address the interesting task of classifying historical texts according to their predicted period of writing. In the digital humanities fields, where many texts contain unidentified or controversial publication dates, this task is useful.

Moreover, in recent years, considerable research has been devoted to diachronic lexical resources, which comprise terms from different language periods [Borin and Forsberg, 2011, Liebeskind et al., 2013, Riedl et al., 2014]. These resources are primarily used for studying language changes and supporting searches in historical domains, thus bridging the lexical gap between modern and ancient languages. Prior work on diachronic lexical resources addressed the problem of collecting relevant related terms from a diachronic corpus, given lexical resource entries. However, before constructing a diachronic lexical resource, a preliminary task of classifying the diachronic corpus documents to historical periods should be confronted.

Typically, two learning approaches for temporal text classification were investigated: supervised and unsupervised. Supervised machine-learning algorithms use the training data of the input examples with their desired output to study a function. Subsequently, the supervised algorithms generalize and predict the value of the function for any unseen valid input [Ciobanu et al., 2013b,a, Štajner and Zampieri, 2013]. In contrast to supervised machine-learning algorithms, unsupervised machine-learning algorithms do not use labeled examples for training. Instead using the data characteristics, unsupervised learning algorithms describe how data are organized or clustered [Liao, 2005, Dalli and Wilks, 2006, Blei and Lafferty, 2006, Wang et al., 2008].

In this paper, we focused on the supervised machine-learning approach. Most conventional supervised machine-learning algorithms for the period classification of historical texts are either rule-based or corpus-based. Their efficiency depends on the prior feature engineering. Studies have shown that given sufficient data, deep-learning models (i.e., deep layers of neural networks) can, without task-specific or data-specific feature engineering and with minimal restrictions, perform classifications on various text genre.

Therefore, we herein investigate deep-learning models for period classification. We compare three different models: paragraph vectors [Le and Mikolov, 2014] (PVs), convolutional neural network (CNN) [Kim, 2014], and recurrent neural network (RNN) [Hochreiter and Schmidhuber, 1997]. The PV model uses paragraph vectors as topic space modeling. The CNN model is a feedforward neural network designed to require minimal preprocessing. Unlike the CNN, the RNN exhibits dynamic temporal behavior, using their internal memory to process arbitrary sequences of inputs.

In our case study, the language of the diachronic corpus is Hebrew. Hebrew had not been used as a spoken language for more than 1,700 years. Different Jewish communities typically spoke local languages or dialects incorporating Hebrew and Aramaic. In contrast, Hebrew was used for formal writing. In the late 19^{th} century and early 20^{th} century with the national renaissance of the Jewish people [Blau, 1981], Eliezer Ben-Yehuda spearheaded the revival of Hebrew as a spoken language [Fellman, 1973]. Ben-Yehuda used 8,000 words from the Bible and 20,000 words from rabbinical commentaries which he codified and integrated into modern Hebrew. Modern Hebrew uses Biblical Hebrew morphemes, Mishnaic spelling, and Sephardic pronunciation, as well as Yiddish idioms.

In addition, the special characteristics of our diachronic corpus render the classification task even more demanding. For example, our corpus of Responsa includes thousands of questions and detailed rabbinic answers on various daily issues. The diachronic Responsa documents present various arguments by citing earlier sources, including the Talmud and its commentators, legal codes, and earlier responses.

In contrast to previous studies that only classified documents as modern or ancient [HaCohen-Kerner et al., 2008, 2010], we distinguished between four periods: the 11^{th} century until the end of the 15^{th} century, the 16^{th} century, the 17^{th} through the 19^{th} centuries, and the 20^{th} century until today.

The application of deep-learning methods to the period classification of Responsa documents revealed interesting results. The CNN and RNN models significantly outperformed the PV and supervised machine-learning algorithms with an accuracy exceeding 81.5%.

This paper is organized as follows: Section I is the Introduction, in Section II, we describe related work on diachronic tasks and the corpus of our case study, the Responsa corpus. In Section III, we provide the necessary background on both conventional machine-learning and deep-learning methods. Subsequently, in Section IV, we compare the results of the three deep-learning models. We also analyzed the semantic changes of words meanings over time in Section V. Finally, in Section VI, we summarize the primary findings and suggests future directions.

II BACKGROUND

This section describes diachronic data and tasks (Section 2.1), followed by a brief introduction to the applications of diachronic tasks on our case study, historical Hebrew Jewish corpus (Section 2.2).

2.1 Diachronic data and tasks

In this section, we focused on two complementary tasks; the first was period classification, or automated dating of documents and the second was identification of word usage changes over time.

Language models were widely used in modeling of temporal information. In an early study, de Jong et al. [2005] used unigram language models combined with smoothing techniques and the log-likelihood ratio measure [Kraaij, 2004] for the automatic dating of Dutch texts (January 1999 to February 2005). Later, the work of de Jong et al. was improved by Kanhabua and Nørvåg [2008] using three new methods: 1) word interpolation for eras a word does not occur, 2) temporal entropy term weighting, and 3) external search statistics.

An unsupervised temporal language model was suggested by Dalli and Wilks [2006] to date texts within a time span of nine years. Their language model is based on temporal-word associations inferred from the time series under the assumption that natural languages feature changing word distributions over time.

More recent, Kumar et al. [2011] followed the language modeling approach. They constructed histograms for a test document based on the Kullback-Leibler divergence between the language model and supervised language models for each era. They demonstrated the effectiveness of their method for predicting the publication dates of short stories, which contain few explicit time references.

Topic models were also investigated for automatic dating of documents. Blei and Lafferty [2006] introduced a discrete dynamic topic model to reveal topic evolution within the time span of a corpus. The model aimed at exploring language changes over time. Subsequent work by Wang et al. [2008] replaced the previous discrete model with a continuous time dynamic topic model.

Until recently, identifying word usage changes over time was challenging due to the lack of electronically formatted historical texts. Google (Google books and Google Ngrams historical projects) has significantly changed this reality owing by providing relevant contextual historical data.

Wijaya and Yeniterzi [2011] conducted an analysis of a Google Books Ngram dataset and found that events occurring in the same era were reflected by contextual word changes. Similarly, Mihalcea and Nastase [2012] showed that significant differences between time periods, on words in context from Google books. Their word samples, divided into three epochs: 1800+/-25 years, 1900+/-25, and 2000+/-25

In addition, Popescu and Strapparava [2013, 2014] investigated the significant changes in the distribution of emotion words and terms. They used the Google N-gram corpus and focused on a statistical approach to infer transition periods between eras with specific characteristics. They observed that the language is not uniformly old or new, and language usage trends and fashions, even for relatively short periods of time (such as 50 years), greatly impact.

Mitra et al. [2015] suggested an unsupervised construct distributional-thesauri-based network method, within a media (Google books data), for data from different eras, which clusters word-centric sense clusters for each era. They also used their method to identify word sense differences and to identify differences in word sense distribution across different media (Google books and Twitter data). They evaluated their method performance both manually as well as through comparison with WordNet.

Google Books Ngrams in languages other than English were also used for temporal text classification. Garcia-Fernandez et al. [2011] introduced a system that predicts the publication dates of excerpts from French journalistic texts. Their system is based on a combination of different systems, relying on both supervised and unsupervised learning and uses several external resources, e.g., Wikipedia, Google Books Ngrams, and etymological background knowledge about the French language.

However, there were studies that focused in languages other than English and did not use Google Books. Ciobanu et al. [2013b,a] applied support vector machine (SVM) and random forest algorithms to classify texts of a historical Romanian text collection. They concluded that the use of lexical features is the best source of data for this task.

Štajner and Zampieri [2013] investigated stylistic changes in a set of Portuguese historical texts from the 17^{th} to the early 20^{th} century, and presented a supervised method to classify them by century. Four stylistic features: average sentence length, average word length, lexical density, and lexical richness contributed to the classification results of 0.92 F-Measure.

Efremova et al. [2015] extended standard classification techniques by combining different models trained on particular eras to obtain optimal time identification in a real-world dataset of more than 100.000 Dutch historical notary acts collected over six centuries. They achieved an overall accuracy above 90% and macro-average indicators above 63%.

More than one language were investigated by Niculae et al. [2014] who proposed a temporal modeling ranking approach for historical texts. Their model can be used to produce reasonable probabilistic estimates of the linguistic era of historical documents in three languages: English, Portuguese, and Romanian.

Four years ago, Popescu and Strapparava [2015] introduced a SemEval diachronic text evaluation task to identify when the piece of news was written. For this task, a corpus of snippets was extracted from a large collection of newspapers published between 1700 and 2010. The evaluation was divided into three subtasks. Each subtask pertained to a specific type of information that might be available in the news. The four systems participating in the tasks have proven that it is an achievable task with interesting possible future research direction.

Frermann and Lapata [2016] modeled word meanings as a set of senses, which are tracked over a sequence of contiguous time intervals, by a dynamic Bayesian model. They applied their model to the SemEval-2015 temporal classification benchmark datasets and demonstrated that the model performance is comparable with highly optimized task-specific systems. However the SemEval-2015 dataset only contains newspapers in English.

Finally, in 2018, Tang [2018] summarized the linguistic-semantic change computation literature. Tang's review identifies five components in the field: diachronic corpus, diachronic word sense characterization, change modeling, evaluation, and data visualization. The review also recognizes the need for more diachronic corpora for languages other than English.

Contemporary semantic change measures include word embedding which measures semantic change patterns based on the lexical similarity of identical words over time [Kim et al., 2014, Kulkarni et al., 2015, Hamilton et al., 2016, Hellrich and Hahn, 2016, Szymanski, 2017]. Words are embedded into a low-dimensional space. A vector of real numbers represents the context of each word. The vectors are meaningless on their own, but semantically similar words exhibit similar vectors. Word embedding is usually done by word2vec models [Mikolov et al., 2013], which are two-layer neural networks trained to reconstruct word in context from a large corpus.

Typically, two protocols for tracking semantic changes have been used: the continuous training model [Kim et al., 2014], where the model for each era is initialized with the embeddings of its predecessor. In contrast, independent training requires mapping between models of different eras [Kulkarni et al., 2015]. Recently, a comparison between these two protocols was performed by Hellrich and Hahn [2016].

In this study, we focus on neural language models for the period classification of historical texts. The experimented models are based on the word2vec word representation, as detailed in Section 3.2.

2.2 The Responsa corpus and diachronic tasks

Our research focuses on the period classification of historical texts from the Responsa project[1]. Our Responsa corpus includes questions and detailed rabbinic answers on many daily issues including law, health, commerce, marriage, education, and Jewish customs. Responsa documents include the Jewish juristic negotiation that resulted in the final rabbinic decision. Responsa documents present various arguments by citing earlier sources, such as the Talmud and its commentators, legal codes, and earlier responses [Koppel, 2011]. Our corpus, used for previous IR and NLP research [Choueka, 1972, Fraenkel, 1976, Choueka et al., 1987, HaCohen-Kerner et al., 2008, Koppel, 2011, Zohar et al., 2013, Liebeskind and Dagan, 2015], contains 76,710 articles and approximately 100 million word tokens. Koppel [2011] emphasized another characteristic of Responsa, Responsa corpus was intended as a source of information and not a source of language use.

Jews are scattered worldwide and the Hebrew language, affected by various local languages, has evolved over the years. Our corpus represents more than a thousand years of global Jewish literary creativity. Therefore, the corpus documents are characterized by various genres and styles. Moreover, the writing style of Hebrew Responsa is enriched by an archaic style. Consequently, the Aramaic citations, primarily from the Talmud, increase the challenge for period classification.

The Responsa project was used by HaCohen-Kerner et al. [2010] to investigate the classification of documents by the ethnicity of their authors and/or to the historical period the documents

[1] http://www.biu.ac.il/jh/Responsa

were written. HaCohen-Kerner et al. applied various machine-learning methods with six sets of stylistic features: quantitative, orthographic, topographic, lexical, function, and vocabulary richness. While the accuracy of the logistic regression method for the ethnicity classification or the period classification was approximately 99.6%, the accuracy of the method for both classification tasks was approximately 98.3%. HaCohen-Kerner et al. classified the Responsa documents into two historical periods: the 19^{th} century and the 20^{th} century.

Recently, methods for automatic and semi-automatic diachronic thesaurus construction in relation to the Responsa corpus were investigated. Thesauri typically contains thousands of entries. Each entry includes a headword and a list of semantically related terms. A diachronic thesaurus enables users to search a modern term and obtain related terms from earlier periods. Thus in a diachronic thesaurus, the headwords are modern and their related terms are ancient.

Zohar et al. [2013] suggested unifying the co-occurrence and distributional similarity approaches for diachronic thesaurus construction. However, Zohar et al. investigated surface-word level without considering item morphology. Consequently, there were many inflections for the same lemma. In the evaluation stage, they classified each related term into lemma based groups. Later, Liebeskind et al. [2012] proposed a schematic methodology for generating a co-occurrence-based thesaurus for the Responsa project. They investigated three options for term representation: surface form, lemma, and multiple lemmas, supplemented with the clustering of term variants.

Liebeskind et al. [2016], using query expansion techniques, introduced an iterative interactive scheme for the construction of a diachronic corpus-based thesaurus. They used the ancient-modern period classification to improve the performance of the statistical co-occurrence measures, and extended their methods for multiword expressions. Liebeskind et al. also classified the candidate terms into two historical periods: ancient (from the 11^{th} century until the 19^{th} century) and modern (from the 20^{th} century).

Mughaz et al. [2015, 2017] investigated how to determine the time-frame a given Responsa author lived. Mughaz et al. [2017] defined a set of key phrases and formulated various rules: iron clad, heuristic, and greedy. Their corpus contains 15,495 files written by 24 rabbis, averaging 643 files for each author. The authorship spanned more than 229 years (1786−2015). The deviation for optimal results for birth year and death year of the Responsa authors was 12.8 years for the authors' year of birth and 9.2 years for authors' death.

III SUPERVISED MACHINE LEARNING FRAMEWORK

Many conventional and deep-learning machine-learning (ML) algorithms have been investigated. Generally, there are three steps in supervised ML algorithms: 1) preparing a training data set with labeled examples. 2) using the labeled dataset, training a classifier (depending on the classification model) may involve feature engineering. 3) predicting the classification of unseen examples.

3.1 Conventional Machine-Learning models

Naive Bayes (NB). NB methods are supervised learning algorithms based on the application of Bayes' theorem with the ”naive” assumption of conditional independence between each pair of features considering the class variable value. While the Multinomial NB implements the NB method for multinomially distributed data, the Multivariate Bernoulli implements the NB method for data that is distributed by multivariate Bernoulli distributions, where each of multiple features is assumed to be a binary-valued variable. NB is a generative classifier that, by estimating the assumptions and distributions of the model, recreates the model that generated the data.

Linear models. Linear classifiers are supervised learning algorithms which make classification decisions based on the value of a linear combination of the features. Discriminative linear classifiers learn the classification from observed data statistics. Two popular discriminative linear classifiers are Logistic Regression (LR) and linear Support Vector Machine (SVM). These methods, using different loss functions, which measures the discrepancy between the classifier's prediction and the true output, both solve the same optimization problem. LR assumes that the observed training set was generated by a binomial model that depends on the output of the classifier and estimates the maximum likelihood of the vector of weight. Whereas, linear SVM maximizes the margin between the decision hyperplane and the examples of the training set.

Multi-layer Perceptron (MLP). MLP is a supervised learning algorithm that learns nonlinear function and classifies linearly non-separable data. MLP has three types of layers: 1) an input layer which consists of a set of neurons representing the input features. 2) one or more hidden layers where each neuron transforms the values from the previous layer with a weighted linear summation, followed by a non-linear activation function, such as the hyperbolic tan function, the logistic sigmoid function, or the rectified linear unit function. 3) an output layer which receives the values from the last hidden layer and transforms them into output values. MLP is a feed-forward neural network model, where the connections between units do not form a cycle. MLP is trained using the backpropagation algorithm Rumelhart et al. [1986].

3.2 Deep-Learning models

Deep-learning (DL) is a subfield of machine-learning based on learning data representations. DL models allow machines to both learn and use the features to perform specific tasks. These models replace the difficult and expensive manual feature engineering processes using domain knowledge data to create features that allow machine-learning algorithms to function.

DL models have demonstrated excellent performances in many NLP tasks. Therefore, we have adopted three state-of-the-art DL models for the period classification of historical texts. In contrast to the conventional MLP deep neural network model trained on bag-of-words vectors, the three DL models were trained using pre-trained word vectors. We provide the relevant background on these models (Section 3.2.1). In Section 4.3, we detail the neural networks architectures we used.

3.2.1 Word Embeddings

Word embedding is the collective name for neural-network-based approaches in which words are embedded into a low-dimensional space. In word embedding models, the context of each word is modeled by a d-dimensional vector of real numbers. The vectors are meaningless on their own, but semantically similar words exhibit similar vectors, and vector similarities are easy to compute.

Mikolov et al. [2013] presented the most popular word embedding models (word2vec) for computing continuous vector representations of words based on extremely large datasets: continuous bag-of-words model (CBOW) and continuous skip-gram model (skip-gram). The models are shallow, two-layer neural networks trained to reconstruct linguistic contexts of words. The CBOW model predicts the target word (w_t) using a context of n words before and after the w_t. The order of context words does not influence the prediction. The skip-gram model is similar to the CBOW model; however, instead of using the surrounding words to predict the center word, skip-gram uses the center word to predict the surrounding words.

Inspired by the methods for learning the word vectors, paragraphs were also mapped to vectors [Le and Mikolov, 2014]. Given the many contexts sampled from the paragraph, the vectors were ”requested” to contribute to the prediction task of the next word.

Two matrices exist in the PV framework: the paragraph matrix with its unique column for every paragraph, and word vector matrix with a unique column for every word. The PV and word vectors are averaged or concatenated to predict the next word in a context. After being trained, the PVs can be used as features to conventional machine-learning techniques including logistic regression, support vector machines, and K-means.

Techniques to adapt unsupervised word embedding to specific applications, when only small and noisy labeled datasets are available have recently drawn research attention. Previous studies have incorporated sentiment information into the neural network to learn sentiment-specific word embeddings [Tang et al., 2014, Zhang and Lan, 2015]. In the past few years, not many studies have learned dynamic word embeddings that capture how the meanings of words change over time [Rudolph and Blei, 2018, Yao et al., 2018]. This interesting research direction of incorporating the period information into the neural network to learn diachronic-specific word embeddings will be investigated in future work.

3.2.2 Convolutional Neural Networks

CNNs are deep, feed-forward artificial neural networks successfully applied to analyze visual imagery. CNN models have recently been shown to be effective for NLP tasks, including part-of-speech tagging, name entity recognition [Collobert et al., 2011], semantic parsing [Yih et al., 2014], sentence modeling [Blunsom et al., 2014], and search query retrieval [Shen et al., 2014].

CNNs consist of an input and an output layer, as well as multiple hidden layers. The hidden layers are either convolutional, pooling, or fully connected. Convolutional layers apply a filter operation to the input, and pass the result to the next layer. Pooling layers combine the outputs of neuron clusters in one layer into a single neuron in the next layer. For example, max pooling uses the maximum value from each of a cluster of neurons at the previous layer. Fully connected layers connect each neuron in one layer to each neuron in another layer.

Kim [2014] reported a series of experiments with CNNs trained using pre-trained word vectors for sentence-level classification tasks (Figure 1).

We applied Kim’s static model directly on our period classification of historical texts task. Their static model is a model with pre-trained vectors from word2vec. All words, including the unknown ones that are initialized randomly, are maintained static and only the other parameters of the model are learned. This simple CNN with little hyperparameter tuning and static vectors yielded excellent results on multiple benchmarks.

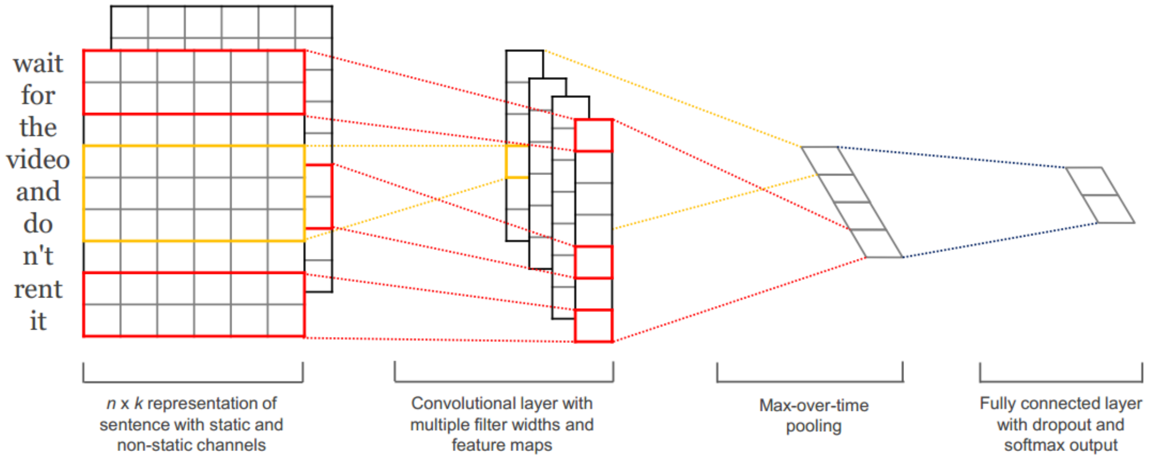
wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 1: Kim et al. [2014] illustration of their model architecture with two channels for an example sentence.

3.2.3 Recurrent Neural Networks

RNN is a class of artificial neural network where connections between units form a directed cycle. RNNs are especially useful for processing sequential data such as written natural language. Recurrent nets differ from feedforward nets because RNNs include a feedback loop, whereby the output from step *n-1* is fed back to the net to affect the outcome of step *n*, and so forth for each subsequent step. In Vanilla RNN, this repeating unit exhibits a highly simple structure, such as a single *tanh* layer. This layer is the RNN hidden layer that modifies the data en route from input to output. The Vanilla version simply calculates the tanh activation function based on the previous hidden state and on the current input. More sophisticated hidden units/layers are used in the RNN models.

Long short-term memory networks (LSTMs) [Hochreiter and Schmidhuber, 1997] are a special type of RNN, capable of learning long-term dependencies. LSTM is typically augmented by recurrent gates called ”forget” gates that allow the unit to decide whether to store or delete information, based on the importance it assigns to the information. The gates are composed of a *sigmoid* neural net layer and a point-wise multiplication operation [Graves and Schmidhuber, 2005].

A popular simpler variation of the LSTM, the gated recurrent unit (GRU) [Cho et al., 2014] model has generated changes in the LSTM unit, including combining the forget and input gates into a single gate and merging the cell state and hidden state.

Lately, combined models of RNN and CNN have been suggested. Tang et al. [2015] introduced convolutional-gated recurrent neural network for document-level sentiment classification. Their model encodes the semantics of sentences and their relations in document representation, and is end-to-end trained effectively with supervised sentiment classification objectives. Tang et al. demonstrated that their models could achieve state-of-the-art performances on four datasets. Lai et al. [2015] also introduced recurrent CNNs to text classification. Their model captures contextual information with the recurrent structure and constructs the representation of text using a CNN. Lai et al. demonstrated that their model outperforms the CNN using four different text classification datasets.

Greff et al. [2017] reported the results of a large-scale study on variants of the LSTM architecture. However, they concluded that the most typically used LSTM architecture [Graves and

Period	Training set	Test set	Total
1	29,107	3,234	32,341
2	50,279	5,587	55,866
3	181,898	20,211	202,109
4	282,620	31,402	314,022

Table 1: Period distribution of the paragraphs in our dataset

Schmidhuber, 2005] can perform reasonably well on various datasets. In the current study, we applied three RNN architectures: the Vanilla, the LSTM, and the GRU, for the period classification of historical texts.

IV EVALUATION

This section presents the evaluation of deep-learning methods for the period classification of historical texts. We first detail the period distribution in our dataset (Section 4.1). Next, we describe our evaluation setting (Section 4.2) and the selected neural networks architectures (Section 4.3) for our experiments. Subsequently, we present the classification results of the deep-learning methods (Section 4.4).

4.1 Period Classification

The Responsa corpus documents can be divided into four periods: the 11^{th} century until the end of the 15^{th} century, the 16^{th} century, the 17^{th} through the 19^{th} centuries, and the 20^{th} century until today. In our experiments, we classified the Responsa historical texts into these four categories.

We divided the history into these four periods since they reflect the historical development of the halakhic (Jewish religious laws) ruling. The first Period (11^{th}-16^{th}) is characterized by the creation of the halakhic codex. The period preceding the publication of the Shulchan Aruch (i.e. Code of Jewish Law), edited by Rabbi Yosef Caro (1563). The second period (16^{th} century) is the period when the halakhic codex began to spread in the Diaspora. The Responsa questions were answered by sages who operated during the reign of Rabbi Yosef Caro. The Shulchan Aruch together with its commentaries is the most widely accepted compilation of Jewish law ever written. The third period (17^{th}-19^{th} century) is the period of the Sages who were already operating after receiving the Halachic Codex (the Shulchan Aruch) as a guiding and significant factor. The fourth period (20^{th} century to the present) is characterized by modernity and scientific and technological development. This period raised entirely new questions and problems in the world of halakha.

One typical barrier for using deep learning to solve problems is the amount of data required to train a model. The large number of parameters in the model that machines have to learn requires much data. To increase the number of texts, we split documents into paragraphs, resulting in a dataset of 604,338 paragraphs. For each period, we randomly selected 90% of the data for training, and the remaining 10% for the test set. Table 1 shows the period distribution of the paragraphs in our dataset. All the paragraphs are at least 10 words in length.

4.2 Experimental setting

4.2.1 Evaluation measures

In our experiments, we compared the performance of the algorithms by four typically used text categorization measures: accuracy, precision, recall, and F1. The scores are macro-averaged; we first calculate the measure for each category and subsequently the average of these scores. Therefore, all our categories were equally weighted. Classes that were never predicted by the model were excluded from the average precision (and F1).

4.3 Neural Networks Architectures

As detailed in Section 3.2, we applied three deep learning methods for period classification. To build the deep neural networks, we used the DL4J (Deep Learning 4 Java) open-source, distributed, deep-learning library for the JVM[2] [Team, 2016] and TensorFlow open-source machine-learning framework[3] [Abadi et al., 2015]. We used these libraries, each with default hyper-parameter settings, as assessed by their authors. The models use 300-dimensional word embeddings, generated by the word2vec skip-gram algorithm with a window size of five words (see Section 3.2.1).

We ran all experiments on a cloud infrastructure using an Ubuntu 16 x64 Virtual Machine, with 32 cores, 128GB Memory and 200GB SSD Disk.

Next, we describe the neural network architecture for each deep-learning method:

Paragraph vectors. We used a PV model to build a paragraph classifier to yield one of four labels; the label with the highest score was used for our classification. The PVs were generated from the training set by the doc2vec algorithm [Le and Mikolov, 2014], which is an extension of the word2vec algorithm where PV and word vectors are averaged to predict the next word in a context (see Section 3.2.1). During training class label features were inserted into the same dense vector space as the words. After training, classification of paragraphs of the test set was performed by calculating the mean vector of the words in a paragraph, and then the paragraph was labelled with a label feature nearest to the mean vector (in terms of cosine similarity). Our learning rate was 0.025 (with a minimal learning rate of 0.001). We trained 10 epochs and the batch size was 100.

Convolutional Neural Network (CNN). We directly applied Kim [2014] CNNs for the sentence classification model, described in Section 3.2.2. Our CNN contains three convolutional layers. A convolution layer involves a filter that is applied to a window of n words to produce a new feature. Our CNN uses three filters (with window sizes of 3, 4, and 5). Subsequently, to capture the most important feature, the model applies a *max-overtime pooling* operation. Finally, a fully connected *softmax* layer exists whose output is the probability distribution over the labels. A Figure of the CNN architecture is presented in Section 3.2.2. We trained the network of one epoch with a batch size of 32.

Recurrent Neural Network (RNN). Our RNN architecture contains two layers: 1) a hidden layer with *tanh* activation function. 2) a *softmax* layer with a *multiclass cross entropy* loss function. Each word in a paragraph was vectorized by word2vec and fed into the RNN. We applied three types of RNN models, 1) Vanilla RNN, 2) LSTM , and 3) GRU, which differs

[2] https://deeplearning4j.org/index.html

[3] https://www.tensorflow.org

Model	Accuracy	Precision	Recall	F1
Multivariate Bernoulli NB	0.7095	0.6639	0.6355	0.6437
Multinomial NB	0.7945	0.7929	0.706	0.7343
LR[1]	0.1924	0.5188	0.5006	0.1982
SVM[2]	0.1233	0.4731	0.4241	0.2047
MLP	0.2129	0.5148	0.5153	0.2211
HaCohen-Kerner et al. [2010]	0.5815	0.5288	0.3215	0.3085

[1] Using absolute norm L1

[2] 1 class excluded from average.

Table 2: Period classification results of the conventional machine-learning models

in the type of the hidden layer (as explained in Section 3.2.3). We trained the network of five epochs with a batch size of 64 and a hidden layer of 128 neurons.

4.4 Results

4.4.1 Conventional Machine-learning methods

To establish a baseline for comparison, we tested the performance of some conventional machine-learning methods (described in Section 3.1) on our dataset. Our dataset includes thousands of examples and a large number of features that do not fit in a computer's primary memory. Thus, we used incremental learning algorithms that learn incrementally from a mini-batch of instances. To perform the incremental methods, we used the scikit-learn open-source machine-learning package in python[4] [Pedregosa et al., 2011]. Because our vocabulary corpus is extremely large (1,406,208 words) and bag-of-words model was used for feature extraction, we selected 20,000 features, words containing at least 10 appearances in the dataset and do not appear in more than 5% of the documents.

We investigated three types of incremental classification algorithms: 1) naive Bayes (NB): multivariate Bernoulli and multinomial, 2) linear models: logistic regression (LR) and linear support vector machine (SVM), and 3) Multilayer perceptron (MLP). Table 2 presents the results of the conventional machine-learning models.

Surprisingly [Ng and Jordan, 2002], the NB methods outperformed the linear models and the MLP algorithm significantly. These discriminative linear and non-linear classifiers classified most of the paragraphs to the first or the second period. It is noteworthy that we examined the two linear models, LR and SVM, with different regularization weights.

To complete our comparison, we tested the performance of HaCohen-Kerner et al. [2010] method which combines conventional machine-learning models with feature engineering. HaCohen-Kerner et al. classified the Responsa documents into two historical periods: the 19th century and the 20th century. They obtained an accuracy of 99.68% using the logistic regression method with four sets of stylistic features:

1. Lexical features: normalized frequencies of 958 common religious words (e.g., bible, Jewish), 533 stopwords, and 307 summarization words (e.g., finding(s), conclusion(s))

[4] http://scikit-learn.org/0.19/modules/scaling˙strategies.html

Model	Accuracy	Precision	Recall	F1	Comments
PV	0.5193	0.3331	0.2511	0.2363	1 class excluded from average
CNN	0.8150	**0.8185**	0.7446	0.7752	
RNN	**0.8495**	0.7958	**0.7747**	**0.7829**	Gated Recurrent Unit (GRU)

Table 3: Period classification results of three deep neural network models: paragraph vectors (PV), CNN, and RNN

2. Orthographic features: normalized number of acronyms in a sentence and normalized number of abbreviations in a sentence.
3. Quantitative features: average number of characters in a word/sentence/paragraph/document, average number of word tokens in a sentence/paragraph/document, average number of sentences in a paragraph/ document, average number of paragraphs in a document, and average number of punctuation marks.
4. Function features: normalized frequencies of sentences appeared in brackets and normalized frequency of 11 non-ambiguous Hebrew pronouns.

The results of HaCohen-Kerner et al. [2010] method on our task are presented in the last row of Table 2. Obviously, classifying documents into four historical periods is much more difficult. Their method achieved an accuracy of 58% which is significantly lower that the NB models.

Previous work reported that the available modern Hebrew morphological analyzing tools have poor performance on the Responsa corpus. For example, whereas the accuracy of a state-of-the-art modern Hebrew tagger on modern Hebrew text was over 90%, on the Responsa corpus it was only about 60% [Liebeskind et al., 2012]. Therefore, HaCohen-Kerner et al. [2010] used the raw text and we followed the same approach.

4.4.2 Deep-learning methods

In this section, we present the period classification results of the deep-learning methods. Table 3 shows the classification results of three deep neural network models: PVs, CNN, and RNN.

The RNN and the CNN models outperform the PV model significantly. Their accuracies (0.8495 and 0.815, respectively) are high. We illustrate the reason for the failure of the PV model by its classification confusion matrix. Each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class. Table 4 shows the confusion matrix of the PV model. The model did not learn how to classify paragraphs from the two first periods correctly. All the paragraphs from these periods were classified as belonging to the third or fourth period. The model also hardly classified paragraphs to one of these periods.

The major reasons for the period classification difficulty are the Hebrew language with the special characteristics of the corpus. Modern Hebrew uses Biblical Hebrew morphemes and Mishnaic spelling increase the difficulty of distinguishing between modern and ancient terms and renders the classification task more difficult. Responsa texts that documents that present arguments by citing earlier sources, such as the Talmud and its commentators, legal codes, and earlier response increases the period classification difficulty add to the already existent difficulties. Distinguishing between an original ancient paragraph and a modern paragraph with earlier citations further compound the difficulty.

In contrast to the PV model, the RNN (with GRU) and the CNN models learned to classify the paragraph to four periods. Table 5 shows the confusion matrix of the CNN model.

Predicted	**Period 1**	**Period 2**	**Period 3**	**Period 4**
Actual				
Period 1	0	0	0	3234
Period 2	0	0	3	5584
Period 3	2	0	282	19927
Period 4	0	0	303	31099

Table 4: Confusion matrix of the paragraph vector model

Predicted	**Period 1**	**Period 2**	**Period 3**	**Period 4**
Actual				
Period 1	2026	443	399	366
Period 2	189	4076	867	455
Period 3	84	444	14074	5609
Period 4	99	104	2123	29075

Table 5: Confusion matrix of the CNN model

To better understand the confusion matrix, we converted the absolute values of the matrix to the relative values. The predicted values of each category were divided by the total number of paragraphs in the category. Table 6 shows the confusion matrix with the relative values. The bold values in the diagonal correspond to the recall of the categories. In italics, for each actual class (row), we marked the most frequent misclassified category. Many of the classification errors were misclassifications of successive periods. This analysis is also valid for the RNN (with GRU) model.

As mentioned in Section 4.3, we applied three types of RNN models: 1) Vanilla RNN, 2) LSTM, and 3) GRU. Table 7 presents the classification results of these three models. The LSTM and GRU models outperform the Vanilla model significantly. It is noteworthy that the LSTM model did not classify any of the paragraphs to the first period. Therefore, we excluded this period from the average precision and F1.

The accuracy improvement of the GRU model over the baseline multinomial naive Bayes model is notable (6 points). The meaning of this result in terms of correctly classified paragraphs is that an additional 3322 paragraphs were correctly classified.

Furthermore, whereas the RNN results can be further improved by parameters tuning or using more sophisticated neural network models [Tang et al., 2015, Lai et al., 2015], considering the independence assumption between the features, the baseline model is limited.

Recently, Yin et al. [2017] systematically compared CNN and RNN on a wide range of representative NLP tasks: sentiment/relation classification, textual entailment, answer selection,

Predicted	**Period 1**	**Period 2**	**Period 3**	**Period 4**
Actual				
Period 1	**0.626**	*0.137*	0.123	0.113
Period 2	0.034	**0.73**	*0.155*	0.081
Period 3	0.004	0.022	**0.696**	*0.278*
Period 4	0.003	0.003	*0.068*	**0.926**

Table 6: Confusion matrix of the CNN model with relative values

Model	Accuracy	Precision	Recall	F1
Vanilla	0.6233	0.4767	0.4645	0.4334
Long Short-Term Memory (LSTM)	0.84	0.7712	0.6493	**0.8112**
Gated Recurrent Unit (GRU)	**0.8495**	**0.7958**	0.7747	0.7829

Table 7: Period classification results of three RNN models: Vanilla, LSTM, and GRU

question-relation matching in Freebase, Freebase path query answering and part-of-speech tagging. Yin et al. concluded that the preferred DL model depends on importance of understanding the entire sequence. They also showed that optimization of hidden layer size and batch size is crucial to good performance of both architectures. Bai et al. [2018] compared CNN and RNN for sequence modeling and showed that CNN outperforms RRN across a diverse range of tasks and datasets. They claimed that CNN should be regarded as a natural starting point even for sequence modeling tasks.

We assume further improvements would be possible by hyper-parameter optimization for both RNN and CNN, but a separate process with a meta-training/validation set split would be required to prevent over-fitting; we did not attempt this. With the default hyper-parameter settings, we observed that the accuracy of the CNN model was 3% higher than the accuracy of the RNN. The CNN is supposed to be effective for extracting position-invariant features and RNN for modeling units in sequence. In our case, the CNN model succeeded in determining the period by some key phrases, better than the RNN model which tried to learn context dependencies.

V SEMANTIC CHANGES OF WORDS MEANING OVER TIME

In this section, we track and analyze the semantic changes of word meanings over time. We adopted the continuous training protocol [Kim et al., 2014] (see Section 2.1), where the model for each time span is initialized with the embeddings of its predecessor. We trained the word vectors using the gensim[5] open-source package [Řehůřek and Sojka, 2010].

5.1 Word Comparisons

To detect words whose usage has modified, we compared the cosine similarity between the same words across our four different eras (Section 4.1).

First, based on the cosine similarity of words in the first period (the 11^{th} century until the end of the 15^{th} century) against their fourth period (the 20^{th} century until today) counterparts, we extracted the most changed words. We omitted words that appear less than 500 times in our dataset. To avoid analyzing words that significantly decreased their usage, we omitted words that occur less than 50 times in each of the periods.

Next, to better understand how these words have changed over time, we compared their neighboring words, based on the cosine similarity, for the first and the fourth periods. The shift meaning of six words from the top 50 most changed words is illustrated in Table 8. The words in parentheses are examples for their neighboring words[6].

The words in Table 8 exhibit different behaviors in the ancient and modern periods. The difference in word behavior may come from changes in sense or changes in wording in the context.

[5] https://radimrehurek.com/gensim/

[6] To facilitate readability, we used a transliteration of Hebrew using Roman characters; the letters used, in Hebrew lexico-graphic order, are *abgdhwzxTiklmns`pcqršt*.

#	Word	Similarity	Ancient meaning (Period 1)	Moden meaning (Period 4)
1	`qr	0.1418	main part (mšm`wt-meaning, mwskm-agreed)	uprooting (`wqr-to uproot, girš-to expel)
2	`cm	0.2061	bone (hirk-the thigh, hcl`wt-the ribs)	essence (`iqr-main, aicwt-quality)
3	whbn	0.2513	and the son (whbt-and the daughter, whab-and the father)	and understand (WDWQ-WDiiq Wmca Ql-search for other sources, qcrti-I shortened)
4	binwnit	0.2646	medium [scale] (mqrq‘-from the ground, mzibwrit-worst quality land)	medium [period] (l‘wnh-to the season, wst-menstruation)
5	xih	0.3096	woman giving birth (iwldt-woman giving birth, ‘qrh-childless woman)	animal (w‘wp-and chicken, aprwx-chick)
6	wmwrh	0.3219	rebellion (swrr-rebellious), teaching (Mhllal-a name of the father of a famous teacher)	leadership (wmnhig-and a leader, wgawn-and a genius)

Table 8: Examples for words that changed their meaning over the years

5.2 Periods of Change

Replicating the evaluation methodology of Kim et al. [2014], to detect periods of change, we draw the time series of a word's distance to its neighboring words from different periods.

Figure 2 shows a graph for the word *`qr* compared to its ancient neighbors, *mšm`wt* and *mwskm*, and the modern neighbors, *‘wqr* and *girš*. Figure 2 illustrates similar graphs for *`cm*, *binwnit*, and *wmwrh*. Such graphs are useful for analyzing the evolution of words. For example, when the typical sense of the word *`cm* (see Table 8, Example #2) started being "essence" and not "bone".

In addition, to recognize a period of change independent of neighboring words, we examined the cosine similarity of a word against itself from a reference period (see Figure 3). Our reference period was the first period. To determine if the change of a word during a given period is different from that expected by chance, we plot the average cosine similarity of all words against their reference points. In Figure 3, we draw the cosine similarity of changed words (as in Figure 2). For comparison, we also show the similarity of unchanged words (based on the cosine similarity of words in the first period against their fourth period counterparts). The unchanged words are as follows: *adwnnw*-our master (0.8766), *dinrim*-dinars (monetary unit) (0.8410), *šaltm*-you asked (0.8289), and *kbwdk*-your honor (0.8285). It is noteworthy that in most of our examples (*`qr*, *`cm*, and *wmwrh*) the major meaning shift was between the second (the 16^{th} century) and third periods (the 17^{th} through the 19^{th} centuries). These meaning shifts may be explained by the revival of Hebrew as a spoken language in the late 19^{th} century.

VI CONCLUSIONS AND FUTURE WORK

In this study, we directed the task of predicting the period when a text was originally written. The period classification of texts is an important preliminary task for diachronic lexical resource construction.

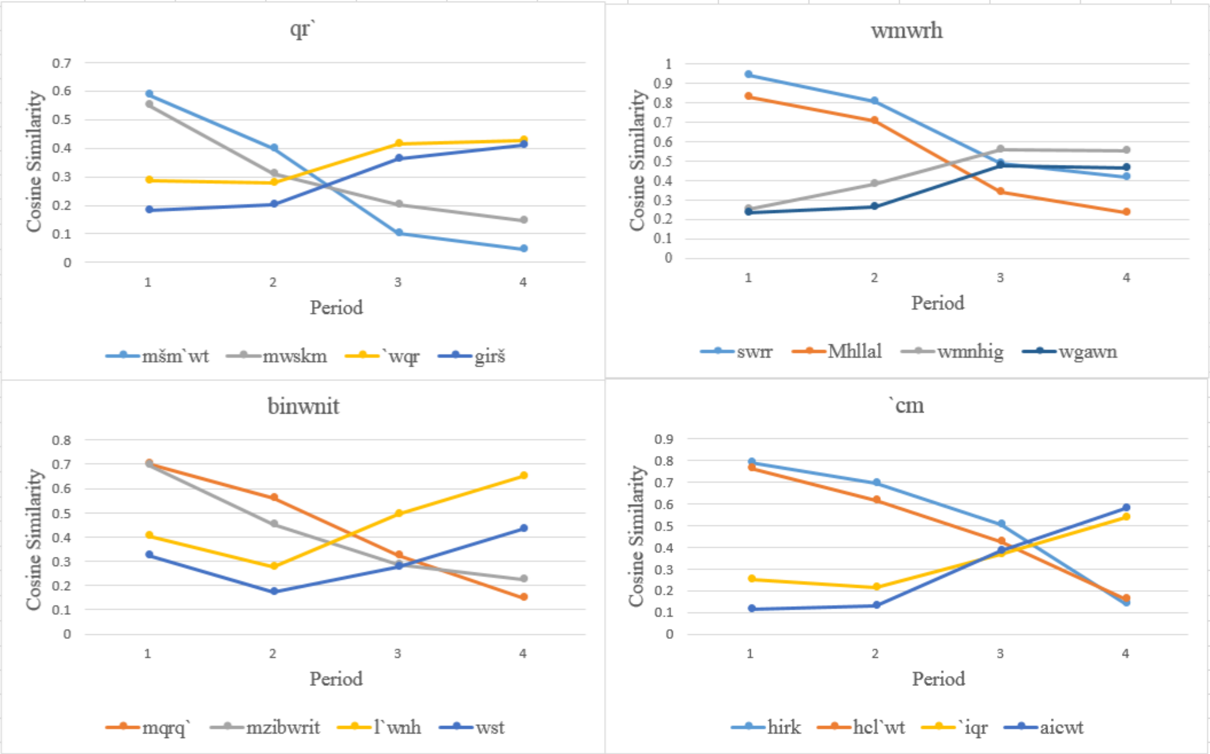

Figure 2: Time trend of the cosine similarity between the words *`qr*, *`cm*, *binwnit*, and *wmwrh* and their neighboring words in the first and fourth periods.

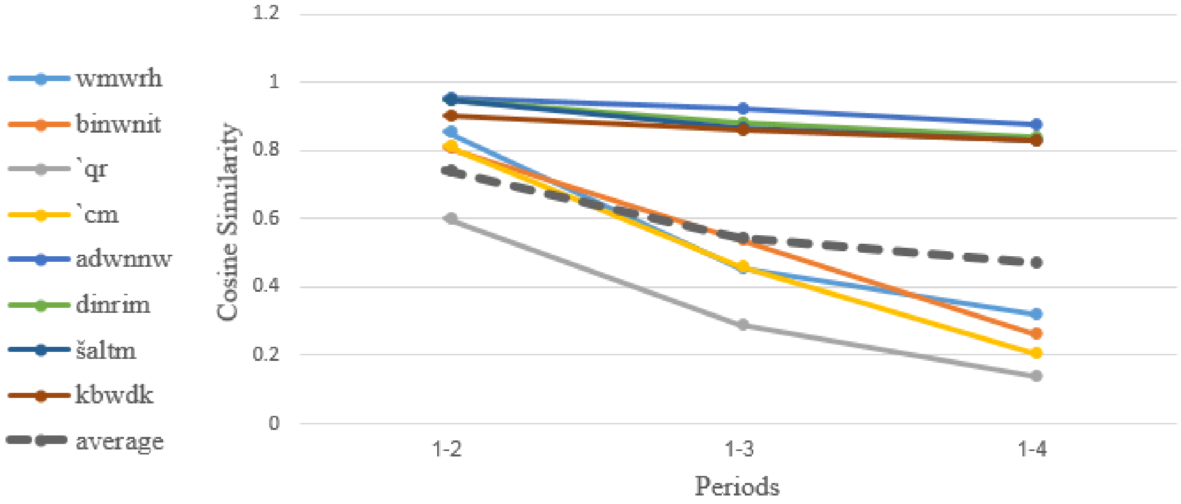

Figure 3: Plot of the cosine similarity of changed and unchanged words against their first period. The broken line is the average cosine similarity of all words against their first period.

While the efficiency of most conventional learning approaches for temporal text classification depends on prior feature engineering, deep learning sets an important goal of automatic discovery of powerful features from raw input data independent of application domains. Thus, we adopted deep-learning algorithms that are supervised machine-learning algorithms inspired by the structure and function of the brain called artificial neural networks.

Our Responsa corpus includes questions on various daily issues posed to rabbis and their detailed rabbinic answers represents more than a thousand years of global Jewish literary creativity. The Responsa documents are divided into four periods.

We compared there typical deep learning methods for the period classification of historical texts: PVs, CNN, and RNN. As baseline, we examined three types of conventional machine-learning methods, NB, linear models, and MLP. The best baseline was NB. However, we observed that the CNN and RNN models outperform both the NB baseline and the PV model significantly. In our context, the accuracy of the CNN model was 3% higher than the accuracy of the RNN. Yet, we assume that further improvements may be possible by hyper-parameter optimization for both RNN and CNN.

We also tracked and analyzed the semantic changes of word meanings over time. We compared the cosine similarity between the same words across our four different time periods to detect words whose usage has changed. Additionally, to detect periods of change, we drew the time series of a word's distance to its neighboring words from different periods and examined the cosine similarity of a word against itself from a reference period.

Our study investigated three RNN models: Vanilla, LSTM, and GRU. The GRU model achieved an accuracy of 84.9%, a recall of 77.47%, and an F1 of 78.29%, and was better that the other two simpler models. We plan to further explore the RNN for period classification by exploring more sophisticated variants of the LSTM architecture [Greff et al., 2017].

In addition, more complex CNN models could be investigated, e.g., the combined models of CNN and RNN [Tang et al., 2015, Lai et al., 2015]. We also propose incorporating the period information into the neural network to learn diachronic-specific word embeddings [Tang et al., 2014, Zhang and Lan, 2015].

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Joshua Blau. *The renaissance of modern Hebrew and modern standard Arabic: Parallels and differences in the revival of two Semitic languages*, volume 18. Univ of California Press, 1981.

David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006.

Phil Blunsom, Edward Grefenstette, and Nal Kalchbrenner. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014.

Lars Borin and Markus Forsberg. A diachronic computational lexical resource for 800 years of swedish. In Caroline Sporleder, Antal van den Bosch, and Kalliopi Zervanou, editors, *Language Technology for Cultural Heritage*, Theory and Applications of Natural Language Processing, pages 41–61. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-20226-1.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Y. Choueka, A. Fraenkel, S. Klein, and E. Segal. Improved techniques for processing queries in full-text systems. In *Proceedings of the 10th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '87, pages 306–315, New York, NY, USA, 1987. ACM. ISBN 0-89791-232-2. doi: 10.1145/42005.42039. URL http://doi.acm.org/10.1145/42005.42039.

Yaacov Choueka. Fast searching and retrieval techniques for large dictionaries and concordances. *Hebrew Computational Linguistics*, 6:12—-32, 1972.

Alina Maria Ciobanu, Anca Dinu, Liviu P Dinu, Vlad Niculae, and Octavia-Maria Sulea. Temporal classification for historical romanian texts. In *LaTeCH@ ACL*, pages 102–106, 2013a.

Alina Maria Ciobanu, Liviu P Dinu, Octavia-Maria Sulea, Anca Dinu, and Vlad Niculae. Temporal text classification for romanian novels set in the past. In *RANLP*, pages 136–140, 2013b.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

Angelo Dalli and Yorick Wilks. Automatic dating of documents and temporal text classification. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 17–22. Association for Computational Linguistics, 2006.

Franciska de Jong, Henning Rode, and Djoerd Hiemstra. *Temporal Language Models for the Disclosure of Historical Text*, pages 161–168. KNAW, 9 2005. ISBN 90-6984-456-7.

Julia Efremova, Alejandro Montes García, Jianpeng Zhang, and Toon Calders. Effects of evolutionary linguistics in text classification. In *International Conference on Statistical Language and Speech Processing*, pages 50–61. Springer, 2015.

Jack Fellman. *The Revival of Classical Tongue: Eliezer Ben Yehuda and the Modern Hebrew Language*. Walter de Gruyter, 1973.

Aviezri S. Fraenkel. All about the responsa retrieval project – what you always wanted to know but were afraid to ask. *Jurimetrics Journal*, 16(3):149—-156, 1976.

Lea Frermann and Mirella Lapata. A bayesian model of diachronic meaning change. *Transactions of the Association for Computational Linguistics*, 4:31–45, 2016.

Anne Garcia-Fernandez, Anne-Laure Ligozat, Marco Dinarelli, and Delphine Bernhard. When was it written? automatically determining publication dates. In *International Symposium on String Processing and Information Retrieval*, pages 221–236. Springer, 2011.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 2017.

Yaakov HaCohen-Kerner, Ariel Kass, and Ariel Peretz. Combined one sense disambiguation of abbreviations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 61–64, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557690.1557707.

Yaakov HaCohen-Kerner, Hananya Beck, Elchai Yehudai, and Dror Mughaz. Stylistic feature sets as classifiers of documents according to their historical period and ethnic origin. *Applied Artificial Intelligence*, 24(9):847–862, 2010.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P16-1141.

Johannes Hellrich and Udo Hahn. An assessment of experimental protocols for tracing changes in word semantics relative to accuracy and reliability. In *LaTeCH 2016—Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities@ ACL*, pages 111–117, 2016.

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

Nattiya Kanhabua and Kjetil Nørvåg. Improving temporal language models for determining time of non-timestamped documents. In *International Conference on Theory and Practice of Digital Libraries*, pages 358–370. Springer, 2008.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1181.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. Temporal analysis of language through neural language models. *ACL 2014*, page 61, 2014.

Moshe Koppel. The responsa project:some promising future directions. In N. Dershowitz and E. Nissan, editors, *Lecture Notes in Computer Science, Springer-Verlag*, Berlin, 2011.

Wessel Kraaij. *Variations on language modeling for information retrieval.* PhD thesis, Neslia Paniculata, 2004.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635. International World Wide Web Conferences Steering Committee, 2015.

Abhimanu Kumar, Matthew Lease, and Jason Baldridge. Supervised language modeling for temporal resolution of texts. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2069–2072. ACM, 2011.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *AAAI*, volume 333, pages 2267–2273, 2015.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

T Warren Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874, 2005.

Chaya Liebeskind and Ido Dagan. Integrating query performance prediction in term scoring for diachronic thesaurus. In *Proceedings of the 9th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, pages 89–94, 2015.

Chaya Liebeskind, Ido Dagan, and Jonathan Schler. Statistical thesaurus construction for a morphologically rich language. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, pages 59–64, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S12-1009.

Chaya Liebeskind, Ido Dagan, and Jonathan Schler. *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, chapter Semi-automatic Construction of Cross-period Thesaurus, pages 29–35. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/W13-2704.

Chaya Liebeskind, Ido Dagan, and Jonathan Schler. Semiautomatic construction of cross-period thesaurus. *Journal on Computing and Cultural Heritage (JOCCH)*, 9(4):22, 2016.

Rada Mihalcea and Vivi Nastase. Word epoch disambiguation: Finding how words change over time. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 259–263. Association for Computational Linguistics, 2012.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Sunny Mitra, Ritwik Mitra, Suman Kalyan Maity, Martin Riedl, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. An automatic approach to identify word sense changes in text media across timescales. *Natural Language Engineering*, 21(5):773–798, 2015.

Dror Mughaz, Yaakov HaCohen-Kerner, and Dov Gabbay. Key-phrases as means to estimate birth and death years of jewish text authors. In *Semanitic Keyword-based Search on Structured Data Sources*, pages 108–126. Springer, 2015.

Dror Mughaz, Yaakov HaCohen-Kerner, and Dov Gabbay. Mining and using key-words and key-phrases to identify the era of an anonymous text. In *Transactions on Computational Collective Intelligence XXVI*, pages 119–143. Springer, 2017.

Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

Vlad Niculae, Marcos Zampieri, Liviu P Dinu, and Alina Maria Ciobanu. Temporal text ranking and automatic dating of texts. In *EACL*, pages 17–21, 2014.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Octavian Popescu and Carlo Strapparava. Behind the times: Detecting epoch changes using large corpora. In *IJCNLP*, pages 347–355, 2013.

Octavian Popescu and Carlo Strapparava. Time corpora: Epochs, opinions and changes. *Knowledge-Based Systems*, 69:3–13, 2014.

Octavian Popescu and Carlo Strapparava. Semeval 2015, task 7: Diachronic text evaluation. In *SemEval@ NAACL-HLT*, pages 870–878, 2015.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

Martin Riedl, Richard Steuer, and Chris Biemann. Distributed distributional similarities of google books over the centuries. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland, May 26-31, 2014.*, pages 1401–1405, 2014.

Maja Rudolph and David Blei. Dynamic embeddings for language evolution. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1003–1011. International World Wide Web Conferences Steering Committee, 2018.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0. URL https://doi.org/10.1038/323533a0.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 373–374. ACM, 2014.

Sanja Štajner and Marcos Zampieri. Stylistic changes for temporal text classification. In *International Conference on Text, Speech and Dialogue*, pages 519–526. Springer, 2013.

Terrence Szymanski. Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 448–453, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/P17-2071.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*, pages 1555–1565, 2014.

Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, pages 1422–1432, 2015.

Xuri Tang. A state-of-the-art of semantic change computation. *Natural Language Engineering*, 24(5):649–676, 2018. doi: 10.1017/S1351324918000220.

Eclipse Deeplearning4j Development Team. Deeplearning4j: Open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0 . 2016. URL http://deeplearning4j.org.

Chong Wang, David Blei, and David Heckerman. Continuous time dynamic topic models. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 579–586, Corvallis, Oregon, 2008. AUAI Press.

Derry Tanti Wijaya and Reyyan Yeniterzi. Understanding semantic change of words over centuries. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 35–40. ACM, 2011.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 673–681, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5581-0. doi: 10.1145/3159652.3159703. URL http://doi.acm.org/10.1145/3159652.3159703.

Wen-tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 643–648, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-2105.

Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

Zhihua Zhang and Man Lan. Learning sentiment-inherent word embedding for word-level and sentence-level sentiment analysis. In *Asian Language Processing (IALP), 2015 International Conference on*, pages 94–97. IEEE, 2015.

Hadas Zohar, Chaya Liebeskind, Jonathan Schler, and Ido Dagan. Automatic thesaurus construction for cross generation corpus. *Journal on Computing and Cultural Heritage (JOCCH)*, 6(1):4:1–4:19, April 2013. ISSN 1556-4673. doi: http://dx.doi.org/10.1145/2442080.2442084. URL http://doi.acm.org/http://dx.doi.org/10.1145/2442080.2442084.