Relating Articles Textually and Visually

Nachum Dershowitz
School of Computer Science
Tel Aviv University
Tel Aviv, Israel
nachum@cs.tau.ac.il

Daniel Labenski
School of Computer Science
Tel Aviv University
Tel Aviv, Israel
daniel.labenski@gmail.com

Adi Silberpfennig
School of Electrical Engineering
Tel Aviv University
Tel Aviv, Israel
adi.silber@gmail.com

Lior Wolf
School of Computer Science
Tel Aviv University
Tel Aviv, Israel
wolf@cs.tau.ac.il

Yaron Tsur
Department of Jewish History
Tel Aviv University
Tel Aviv, Israel
ytsur@tau.ac.il

***Abstract*— Historical documents have been undergoing large-scale digitization over the past years, placing massive image collections online. Optical character recognition (OCR) often performs poorly on such material, which makes searching within these resources problematic and textual analysis of such documents difficult.**

We present two approaches to overcome this obstacle, one textual and one visual. We show that, for tasks like finding newspaper articles related by topic, poor-quality OCR text suffices. An ordinary vector-space model is used to represent articles. Additional improvements obtain by adding words with similar distributional representations.

As an alternative to OCR-based methods, one can perform image-based search, using word spotting. Synthetic images are generated for every word in a lexicon, and word-spotting is used to compile vectors of their occurrences. Retrieval is by means of a usual nearest-neighbor search. The results of this visual approach are comparable to those obtained using noisy OCR.

We report on experiments applying both methods, separately and together, on historical Hebrew newspapers, with their added problem of rich morphology.

I. Introduction

Recent large-scale digitization and preservation efforts have made a huge quantity of images of historical books, manuscripts, and other media readily available over the Internet, opening doors to new opportunities, like distant reading and data mining.[1]

Research supported in part by Grant #I-145-101.3-2013 from the German-Israeli Foundation for Scientific Research and Development (GIF), Grant #01019841 from the Deutsch-Israelische Projektkooperation (DIP), Grant #1330/14 of the Israel Science Foundation (ISF), and a grant from the Blavatnik Family Fund. It forms part of D.L.s and A.S.'s M.Sc. theses at Tel Aviv University.

[1] http://www.nytimes.com/2011/06/26/books/review/the-mechanic-muse-what-is-distant-reading.html.

The task we tackle is topic detection – specifically, looking for newspaper articles that discuss the same ongoing event. Identifying related articles in historical archives can help scholars explore historical stories from a broad perspective, can provide visitors of archival websites with effective navigation – as in modern news sites, and can help compare the coverage of events between different publications.[2] Unfortunately, optical character recognition (OCR) for older documents is not yet satisfactory, making it quite challenging to search within those images for something particular, limiting research options.[3]

We adopt the following nomenclature [1]: An *event* is "something that happens at a specific time and location". A *story* is "a topically cohesive segment of news that includes two or more declarative independent clauses about a single event"; in our case this is always a single newspaper article. A *topic* is "a set of news stories that are strongly related by some seminal real world event". We use *subject* to refer to thematically related topics. For example [2], the eruption of Mount Pinatubo on June 15th, 1991 is an event, whereas the ensuing crisis and the aftermaths caused by the cataclysmic event are part of the larger topic, and the subject is "natural disasters". The following is one of the stories that day (*New York Times*):

[2] A list of archives may be found at https://en.wikipedia.org/wiki/Wikipedia:LOONA.

[3] See, for example, the famous newspaper article by Craig Claiborne (*The New York Times*, November 14, 1975), "Just a quiet dinner for two in Paris" (https://ia801009.us.archive.org/18/items/354644-claiborne/354644-claiborne.pdf, last item), which read as gibberish not too long ago (https://assets.documentcloud.org/documents/354644/claiborne.txt).

> Mount Pinatubo rumbled today with explosions that hurled ash and gas more than 12 miles high.... President Corazon C. Aquino dismissed a British newspaper report that the Americans had warned her of possible radioactive contamination if the volcano damaged nuclear storage sites at Clark Air Base.... She said the story was "baseless and purely fabricated." ...

We first consider the case where printed texts have been digitized, resulting in noisy OCR text. We approach the problem traditionally and then enhance it by using word embeddings to augment document representations. This helps solve the sparsity problem, due to language variation, morphology, and OCR noise. Augmentation proved to be helpful in finding more related articles and proved to be even better than language-specific standardization methods.

As an alternative to working with OCR results, we propose a novel image-based approach for retrieval. Given a query image of a word, one seeks all occurrences of that same word within the dataset. To that end, we utilize a word-spotting engine that is both simple to implement and fast to run, making it extremely easy to incorporate for a wide range of resources. The method is completely unsupervised: words in the document images are unsegmented and unlabeled.

The next section summarizes some recent approaches to topic detection and to word spotting. It is followed by a section on the newspaper corpus we experimented with, its problematics, and the additional linguistic/visual resources and tools that we used. Section IV describes the textual method, and Section V describes the visual one. They are followed by experimental results for the two methods. We conclude with a brief discussion.

II. Related Work

Topic Detection: The task we study is related to the field of topic detection and tracking (TDT). This research program, which began in 1997, aims to find core technologies for organizing streams of news articles by topic [3]. Event detection has recently regained popularity with the emergence of social media [4], [5]. We look for stories in a retrospective fashion on a static database (Retrospective Event Detection [RED]), unlike TDT, where one wishes "to identify new events as they occur, based on an on-line stream of stories" (New Event Detection [NED]). For example, a common way of approaching detection is by representing stories via a set of features and assigning every new story to the cluster of the most similar past one. Yang et al. [6] found that retrospective event detection can obtain much better results than the online setup.

Vector Representations: For solving such tasks, it is common to represent each story as a vector in a conventional vector space [3]. Various term weighting schemes for combining *term frequency (tf)* and *inverse document frequency (idf)* can be employed [7]. We use this standard method for document representation and measuring similarity.

Allan and Harding [2], [8] used a story representation of tf-idf weights of the 1000 most weighted terms in the document, computed cosine similarity of the story to every previous story and assign a story to the nearest neighbor cluster if the similarity was above a threshold otherwise it creates a new cluster containing the story. We take this basic approach. Other document representation and similarity measures have been suggested; for example, [9], [10] use language models for clusters to compute the probability that a new story is related to a cluster and use that as a similarity measure.

Lexical Expansions: Petrović and Osborne [4] use lexical paraphrases from various sources of paraphrases, including WordNet, to expand document vectors and overcome lexical variation in documents with different terms discussing the same topic. Moran and McCreadie [5] show how to use word embedding to improve accuracy by expanding tweets with semantically related phrases. Their method does not depend on complex linguistic resources like WordNet and still gets better results. This resource independence allows one to run it on low-resource languages. We use a similar approach to augment the document bag of words.

Morphological Expansions: Avraham and Goldberg [11] reported that, when using word embedding in Hebrew, which is a morphologically rich inflectional language, word vectors bear a mix of semantic and morphological properties. Many neighboring word vectors are of inflected forms, rather than having semantic similarity. We will leverage morphological properties of word embeddings to expand terms with others sharing the same base form.

Noisy Texts: Agarwal et al. [12] experimented with texts from different resource types in order to understand how different levels of noise in the text harm text classification success. They developed a spelling error simulator and created synthetic texts with several levels of noise. They found that with up to 40% word error rate, the accuracy of the classification was not significantly affected. Vinciarelly [13], [14] reported similar results in text clustering and categorization (classification) tasks. We were encouraged by these results to work on topic detection despite the noisy OCR text.

Word Spotting: Whereas topic tracking in the field of NLP is widely researched, we suggest here a novel approach using images as sole input. This appears to be the first attempt to take an image-based approach for solving the topic-detection problem. The work of Wilkinson et al. [15] is somewhat related. They use a word-spotting engine for computing word clouds, whereas we are applying spotting to topic detection and categorization.

Much progress has been made throughout recent years in the area of word spotting. We employ the method of [16] (see Section III); other recent methods include [17], [18], [19], [20], [21], [22], [23], [24].

III. Resources

Corpus: Our experiments were performed on texts from the Jewish Historical Press (JPress),[4] which offers full-text search of 124 publications (1.6 million pages) in 10 languages from a wide variety of Jewish communities. Newspapers were digitized, segmented into articles (with manual help), and processed by an OCR engine. Even though these technologies are mature and used heavily in many applications, they are not foolproof. JPress has to contend with certain phenomena that reduce the success of OCR and segmentation:[5] "These phenomena include inferior quality of printing (which characterizes early publications), yellowing paper, marks or damage in the original printing, unique fonts, torn pages, scribbled-on pages, and even pages damaged by rodents."

Technological limitations and poor quality raw materials lead to word identification problems, where a word is not identified, or is identified incorrectly (characters were replaced or added), or words that were derived from noise in the image do not really exist. Another problem is incorrect segmentation, where a purported article contains more than one real article or one real article is split into multiple ones. We will ignore the latter problem and regard a returned result as correct if any component of the segment matches.

Test Suite: The methods were evaluated primarily on a dataset comprising all issues of *Davar* from February 1981,[6] for a total of 10881 images, divided into 3233 articles. We chose 111 query articles, some reporting a major continuous event, while others reporting on a singular event with few or no related articles.

The average word error rate for this newspaper is about 20% [Ido Kissos, private communication], but is somewhat worse for the test suite. (For example, the error rate in the first two paragraphs of the lead article of the February 1 issue is $30/85 = 35\%$, and may even be considerably worse.) Only about 45% of the tokens produced by OCR of the test corpus are found in the MILA lexicon of modern Hebrew,[7] on account of OCR errors, inflections, named entities, etc.

Lexicon: We make use of a Hebrew corpus (MILA) containing the protocols of Knesset sessions from January 2004 through November 2005.[8] We compiled a list of all tokens appearing 10 or more times, removing non-Hebrew words. Out of these, we took the most frequent 100,000 words as our lexicon. For a visual lexicon, these were rendered as synthetic images in a standard font resembling most of the newspapers in the corpus.

Morphological Tools: Hebrew is a morphologically-rich language, with prefixes and suffixes attached to lemmas. Accordingly, we employ a Hebrew morphological analyzer, HebMorph,[9] that associates a lemma to each word form found in the hspell word list [25].

Representation Tools: Distributional word representation algorithms utilize the contexts in which words appear and their co-occurrence distributions to represent words. Their output is a mapping from words (and phrases) to real-number vectors. Recently, Mikolov et al. [26] presented a word-embedding tool, called "word2vec", that uses shallow neural networks.[10] It has gained popularity due to its efficiency and its success in representing words as vectors, which has been used in many NLP tasks to obtain state-of-the-art results.

Word Spotting: For the visual method, we try to find all occurrences of each word in the lexicon. We used an efficient word-spotting engine from [16], inspired by the work of Liao et al. [27] in the domain of face recognition. Training data is unsupervised: no classifiers are used to learn similarities. The process starts by binarizing the newspaper images and forming bounding boxes around candidate target words. As the images are not pre-segmented into words, candidate words are oversampled. Each target is resized to a fixed-size rectangle and represented by image descriptors, followed by max-pooling. After spotting, only the best of multiple overlapping results is chosen.

Indexing the words of each newspaper page takes a few seconds, depending on the page's complexity.

[4] http://jpress.org.il (Hebrew); http://web.nli.org.il/sites/JPress/English (English).

[5] http://web.nli.org.il/sites/JPress/English/about/Pages/about_technology.aspx.

[6] http://web.nli.org.il/sites/JPress/English/Pages/Davar.aspx.

[7] http://mila.cs.technion.ac.il/resources_lexicons_mila.html.

[8] http://mila.cs.technion.ac.il/resources_corpora_haknesset.html.

[9] https://github.com/synhershko/HebMorph.

[10] https://code.google.com/archive/p/word2vec.

Retrieval requires a fraction of a second. Though this may not be the most accurate of spotting technologies, it is certainly one the simplest and provides a level of accuracy within reach of state-of-the-art ones. Specifically, no optimization is performed during preprocessing or during querying, representations are easily stored in conventional data structures, and the whole pipeline is completely unsupervised. This contrasts with methods that employ techniques such as neural networks, SVM or DTW.

IV. Textual Method

Given an article in a (historical) newspaper, we wish to find more articles relating to the same event(s). We do this by representing articles as vectors and finding articles that have the closest vector to that of the query.

The main steps are the following: (A) text cleaning and preprocessing; (B) representing articles as vectors; (C) finding nearest neighbors of query vectors.

A. Text Preprocessing

OCR outputs many non-alphanumeric characters because of the noise in the image that is sometimes mistaken for numbers or the like. These spurious characters are unlikely to appear again in the same word exactly in the same location, and they cause the article vectors to become much sparser and harder to compare. So, we remove all non-alphanumeric characters, including punctuation, which has no impact on the semantics of the article. (In Hebrew, the only actual words affected by the removal of punctuation are acronyms that have a double quotation mark in the penultimate position to signify the abbreviation, but all such acronyms will map to the same words, and it is relatively rare for ambiguity to be introduced as a consequence.) Another problem that is common to OCR systems is poor word segmentation. Some words are concatenated, while sometimes the output contains single letters as words—usually because of noise in the image. We remove all single-letter words, as there are none in Hebrew. On one particularly bad example, this *cleaning* process reduced the word error rate from 82% to 54%.

B. Representing Articles

Set of Words: This is the simplest baseline. Every document is represented by an unordered collection of its distinct words, and so the dimension of each binary vector is the same as the size of the vocabulary (number of distinct words occurring in the corpus). In this scheme, all words in a document get the same weight regardless of how many times each appears in the document and corpus.

Tf-Idf: This well-known representation is standard in information retrieval [7]; while simple, it gives good, competitive results. We used the square-root normalized version of tf-idf, as it performs well.

Morphological Augmentation: Adding lemmata to the vectors, while still keeping the original word, can increase the hit ratio when comparing documents. We use the Hebrew morphological tool to enrich vectors in this way. In case a token isn't recognized, we try to tolerate spelling errors and orthographical variants by deleting *matres lectionis* (optional vowel letters).

Augmenting with Word2vec: We leverage two attributes of word embeddings: words with different affixes but with a shared lemma have similar vectors; and words with OCR recognition errors also have vectors that are similar to those of the actual words. The first attribute helps achieve generality similar to that of a morphological analyzer but without any language-specific knowledge. The second can help counter the noise in the OCR output, but – unlike OCR post-processing for correcting errors – we do not need to replace the error with the correct word but rather with a good representative that appears in other documents.

After running word2vec on the cleaned OCR'ed text, the following is performed for each word in the article:

1) Find the k nearest neighbors of the word (we used $k = 20$).
2) Filter the neighbors with edit distance less than d (we used $d = 3$).
3) Add to the article's bag of words the most frequent neighbor if its frequency is greater than the original word's.

We trained word2vec over all of the articles in the corpus, using word2vec's default configuration: continuous bag of word architecture with dimensionality of the word vectors set to 200, context window size set to 5, and the hierarchical softmax training algorithm.

Since we are matching stories about specific related events, not broad categories, we wanted to augment the documents' vectors only with syntactically similar words and not with semantic paraphrases. So we filter out all neighbors that are at a large (standard) Levenshtein edit distance. Tests showed that 3 (additions/deletions/substitutions) is a good threshold, even though there still are a few cases where semantic neighbors (not sharing the root) pass through this filter.

Word2vec augments the vector with many mistaken OCR readings, as well as with words that have the same lemma, whereas the morphological analyzer only helps with the latter. Another advantage of word2vec

is in augmenting named entities, which are frequent in the corpus, but which the lexicon does not cover and the analyzer cannot handle. The augmentation size of the Word2vec method per word is 0 or 1, whereas the morphological augmentation average size is 3, making the vector more compact and efficient while getting better results.

Word-Vector Averaging: This method is a common baseline in sentence and document vector representation. It gives the centroid of a document's distributional word vectors, and is used to represent a document's topic. We found that using tf-idf weights for the weighted arithmetic mean of the word vector performs better than uniform weighting, and therefore only report results with frequency weighting.

C. Finding Neighbors

We use cosine similarity, popular in topic detection experiments [28], [29].

V. Visual Method

As in the textual approach, each article is represented as a vector, and one computes the distance between a query and all the articles in the dataset. Whereas when assuming access to the text it is quite straightforward to represent articles as vectors, we work now only with images as input. Since we cannot use the text itself to build a word list, we created a visual lexicon.

A. Method Description

We used the pre-computed Hebrew lexicon and generated synthetic images for each of its words. These synthetic images serves as queries and were fed to the word-spotting engine, which searched for visual matches within the images of the corpus. Based on these spotting results, we represent each article in the dataset using a term vector model combined with a traditional tf-idf weighting scheme. Finally, we retrieve related articles using cosine similarity for the nearest neighbor search, as in the textual method.

First we run word spotting to extract candidate words from the corpus, using the engine of [16]. The parameters can be divided into two groups: those concerning word detection in a binarized image and those that govern the encoding process. The specific settings for the word detection were chosen based on a small sample: Connected components for candidate letters were in the range $[35\,..\,2000]$ pixels. The allowed height range was $[4\,..\,120]$ pixels, and width was $[3\,..\,280]$. For candidate words, the bounding boxes' minimal height and width were 20 and 13 pixels, respectively. The maximal inter-word gap was 32 pixels and the maximal interline gap, 12. Candidates had a maximum bounded rectangle of 240×55 pixels. We also applied jittering as described in [16] and the candidate-extraction stage resulted in 2,289,263 word candidates. The number of actual words is considerably less due to both jittering and the over-compensating detection process that suggests many word candidates for every real word.

The encoding pipeline resized each word candidate to a fixed 160×56 pixels, divided into a grid of 20×7 cells, each of 8×8 pixels. We used max pooling with a random partition of 3750 random exemplars into 250 subsets, each of cardinality 15. Each word is therefore represented by a vector in $\mathbb{R}_{+}^{250}$ containing pooled distances to random exemplars. This matrix containing the encoding for all word candidates fits in RAM and allows for efficient queries.

Each query image from the visual lexicon is considered separately, represented in the same way as candidate words. We tried different font sizes, and empirically found that large font sizes produce better spotting results, and therefore chose a size of 100pt. The query vector is compared to the similar vectors of all candidate targets extracted from the dataset. All candidates are ranked in accordance with the computed $L2$ distance. Then, for each query, we apply a threshold and consider only the set of candidates for which the visual similarity between the corpus candidate and that of the query passes a threshold τ. (The default was $\tau = 0.99$.)

For better accuracy, we also disallow any candidate word image from appearing in more than one set of word matches. Out of all matches for the same bounding box, only the word with the highest rank is considered.

B. Article Representation

Applying a threshold enables us to count lexical occurrences in documents and treat our problem as a NLP task. As is usual, we use tf-idf. For term frequency, we compared three common variants: raw frequency of a term in a document, logarithmic-scaled frequency, and square-root of frequency. The inverse document frequency is the logarithmic scaled inverse fraction of the document that contain the term.

Given an article, it is treated as explained above and represented as a vector using the tf-idf method, resulting in a vector in $\mathbb{R}^m$, where m is the size of the lexicon. Nearest neighbor search is then performed, using the cosine measure. The results are ranked in descending order and the k-top results are returned.

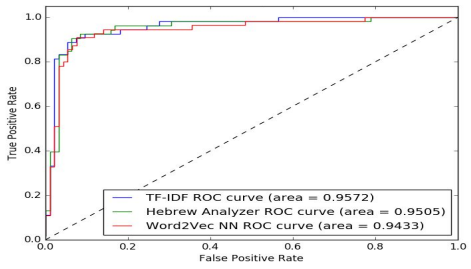

Fig. 1: ROC curve for locating parallel articles.

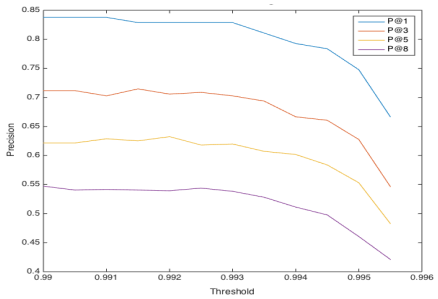

Fig. 2: The sensitivity of topic detection to the threshold τ. The graph shows precision while varying the threshold τ used to decide whether a spotted result is accepted.

VI. Results

We evaluate the two methods, noisy OCR text and word spotting, on the same task of finding articles describing the same topic from different editions of the same newspaper.

The test is comprised of 111 articles chosen as queries, with the goal of retrieving all other items in the 3233-article test suite that refer to the same topic. For each query, the top 8 results were checked manually and labeled as one of same topic, same subject (but not same topic), or unrelated. The average number of related events per query is 9.4 and the median is 7.

To measure recall, one needs a gold standard, listing *all* relevant articles for each query. The "gold" standard used below must be taken with a grain or more of salt, as it comprises all relevant articles found during any of the textual and visual experiments, but may miss a few relevant articles that never showed up.

A. Textual Method

As can be seen in Table I, the basic use of tf-idf vectors got about 90% of the related articles identified in the gold standard. The vectors augmented with Hebrew stems improved the results by another 2%, while the word2vec augmentation added 3% to the tf-idf baseline, beating the Hebrew analyzer. (Occasionally, there was no follow up story, which is why precision even at rank 1 need not be perfect.)

Unsurprisingly, vector centroids did not give good results; they did, as should be expected, return articles on the same subject, rather than the same topic.

We performed additional tests on finding parallel articles (same day) in another newspaper (*Maariv*[11]) in the JPress archive. Of 59 events that appeared in both papers, tf-idf alone gave a precision of 91.5% for the top match; augmenting with word2vec gave 93.2%. The ROC curves for this experiment are shown in Fig. 1.

[11] http://web.nli.org.il/sites/JPress/English/Pages/maariv_test.aspx.

B. Visual Method

Since our goal is to compare approaches, we report precision at k for the visual method, alongside the results of the OCR-based method. Table II shows the precisions achieved by the visual detection system and its variants. As can be seen, normalization of tf-idf gives a sizable improvement compared to the baseline tf-idf. The log normalization variant seems to perform slightly better than root normalization (used in the textual method) and gives reasonable accuracy compared to the text-based approach, retrieving 84% of the possible related articles in the gold standard at rank 1, compared to 87% for the textual method. Overall, the textual method slightly outperforms the visual, across the board. See Table III.

We also present the results for the categorizing by subject in a second column. As can be seen, for categorization the gap between the visual and text-based methods is much reduced.

We studied the effect of the visual-similarity threshold τ, which we integrated in the word-spotting engine, on both topic detection and categorization. Fig. 2 shows the effect on topic detection using log normalization. The systems are relatively robust with regard to this parameter. Within the range $[0.990\,..\,0.993]$, the dependency on the threshold is quite flat, and although there are many false positives, precision is high. Once τ goes above 0.993, not enough matches are accepted, so the number of false negatives is large and performance starts to degrade.

We also tried using the Hebrew morphological analyzer to augment terms, as we did in the textual experiments, but there was no noticeable improvement.

C. Combined Method

Combining the two methods, giving equal weight to the textual and visual vectors, after normalizing the

TABLE I: Textual method for topic detection.

Method	Precision@1	Precision@3	Precision@5	Precision@8	Recall@8
Baseline (set of words)	0.77	0.66	0.58	0.48	0.54
Tf-idf (square root)	0.87	0.76	0.70	0.62	0.64
Word vector centroid	0.73	0.62	0.52	0.43	0.48
Morphological augmentation	0.89	0.78	0.71	**0.65**	**0.73**
Word2vec augmentation	**0.90**	**0.80**	**0.72**	0.64	0.72
Gold standard	0.96	0.89	0.82	0.71	0.79

TABLE II: Visual method for topic detection.

Method	Precision@1	Precision@3	Precision@5	Precision@8	Recall@8
Tf-idf	0.78	0.67	0.58	0.52	0.53
Tf-idf root normalization	0.83	0.70	0.62	0.54	0.55
Tf-idf log normalization	**0.84**	**0.72**	**0.64**	**0.56**	**0.57**
Gold standard	0.96	0.90	0.85	0.75	0.79

TABLE III: Comparative results for topic and subject detection.

Method	Precision@1		Precision@3		Precision@5		Precision@8		Recall@8	
	Topic	Subject	Topic	Subject	Topic	Subject	Topic	Subject	Topic	Subject
Visual	0.84	0.98	0.72	0.94	0.64	0.91	0.56	0.89	0.57	0.33
Textual (no augmentation)	0.87	0.99	0.76	0.95	0.70	0.93	0.62	0.89	0.64	0.34
Gold standard	0.96	1.00	0.90	0.98	0.85	0.94	0.75	0.90	0.79	0.38

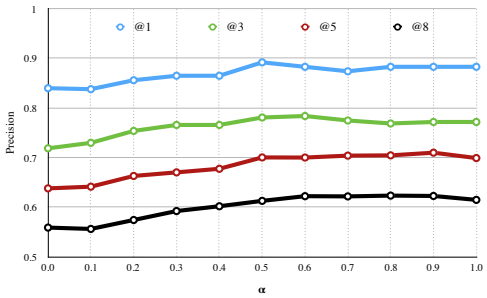

Fig. 3: Combining the methods, ranging from the purely visual on the left ($\alpha = 0$) to purely textual on the right ($\alpha = 1$), and measuring Precision@k, for $k = 1, 3, 5, 8$. The textual vector is given weight α and the visual, $1-\alpha$.

distribution of scores of each method, makes for a small improvement in precision of the top 1 and 3 results. The combined method oftentimes promotes one additional correct match. See Fig. 3.

VII. Discussion

We have explored different ways of enriching historical datasets with language-based tools that can be incorporated in utilities to help in studies depending on the huge mass of historical information now available in digital form. We showed how historical Hebrew newspaper articles can be connected together into a continuous story with high precision, despite poor quality OCR.

To find related articles in the OCR'ed newspapers, we use a traditional tf-idf vector representation, which finds related articles by using nearest neighbor search with a cosine similarity measure. It proved to work well even though the input was very noisy and the language highly inflectional and morphologically rich. We found that affix removal and augmentation with lemmas improves precision. More importantly, we showed how to achieve even better results without any language specific resources by using word embedding similarities.

This suggests that word embeddings should be utilized for query expansion also in other information retrieval tasks, and could be used for term standardization instead of the usual lemmatization. For noisy OCR texts, this method can be used for post-processing error correction, and also to help find named entities that are not read correctly by OCR.

As an alternative to working with noisy text – so long as quality OCR technologies are still lacking for dealing with handwritten manuscripts and historical newspapers and documents, we have demonstrated that word-spotting technologies are up to the challenge.

Our visual alternative does not require any labeling nor ground truth and has the advantage that document

images are its sole input and OCR is not needed at all. We have seen the feasibility of this image-based approach, and showed that it performs almost as well as a text-based system. On the other hand, the more we expand its vocabulary, with named entities for example, the better we should expect the visual method to perform.

We used an efficient and relatively flexible underlying word-spotting engine. This flexibility implies that word-spotting can be very useful in practice – either as is, or as a plug-in method that can provide a list of candidates to another system. Since no optimization is necessary and the whole process is unsupervised, it can easily be adapted to different types of datasets of imaged texts.

Finally, we have seen that categorization works well, too, which suggests first running a classifier and using only subject-related articles for topic and event detection.

REFERENCES

[1] R. Nallapati, A. Feng, F. Peng, and J. Allan, "Event threading within news topics," in *Proc. Thirteenth ACM International Conference on Information and Knowledge Management (CIKM '04)*. New York, NY: ACM, 2004, pp. 446–453.

[2] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic detection and tracking pilot study – Final report," in *Proc. Broadcast News Transcription and Understanding Workshop*, 1998, pp. 194–218.

[3] J. Allan, *Introduction to Topic Detection and Tracking*. Boston, MA: Springer, 2002, pp. 1–16.

[4] S. Petrović, M. Osborne, and V. Lavrenko, "Using paraphrases for improving first story detection in news and Twitter," in *Proc. 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT '12)*. Stroudsburg, PA: Association for Computational Linguistics, 2012, pp. 338–346.

[5] S. Moran, R. McCreadie, C. Macdonald, and I. Ounis, "Enhancing first story detection using word embeddings," in *Proc. 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. New York, NY: ACM, 2016, pp. 821–824.

[6] Y. Yang, T. Pierce, and J. Carbonell, "A study of retrospective and on-line event detection," in *Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. New York, NY: ACM, 1998, pp. 28–36.

[7] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, Aug. 1988.

[8] J. Allan, S. Harding, D. Fisher, A. Bolivar, S. Guzman-Lara, and P. Amstutz, "Taking topic detection from evaluation to practice," in *Proc. 38th Annual Hawaii International Conference on System Sciences (HICSS'05) - Track 4 – Volume 04*. Washington, DC: IEEE Computer Society, 2005, pp. 101a–101a.

[9] J. P. Yamron, S. Knecht, and P. Van Mulbregt, "Dragon's tracking and detection systems for the TDT2000 evaluation," in *Proceedings of Topic Detection and Tracking Workshop*, 2000, pp. 75–80.

[10] M. Spitters and W. Kraaij, "Using language models for tracking events of interest over time," in *Proc. Workshop on Language Models for Information Retrieval (LMIR) 2001*, 2001.

[11] O. Avraham and Y. Goldberg, "Word embeddings in Hebrew: Initial results," in *Israeli Seminar on Computational Linguistics (ISCOL)*, Raanana, Israel, 2015.

[12] S. Agarwal, S. Godbole, D. Punjani, and S. Roy, "How much noise is too much: A study in automatic text classification," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, Oct 2007, pp. 3–12.

[13] D. Grangier and A. Vinciarelli, "Noisy text clustering," Idiap Research Institute, Martigny, Switz., Tech. Rep. IDIAP-RR 04-31, Dec. 2004.

[14] A. Vinciarelli, "Noisy text categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1882–1895, 2005.

[15] T. Wilkinson and A. Brun, "Visualizing document image collections using image-based word clouds," in *Proc. 11th International Symposium on Advances in Visual Computing (ISVC), Part I, Las Vegas, NV*, ser. Lecture Notes in Computer Science, vol. 9474. Cham, Switz.: Springer, 2015, pp. 297–306.

[16] A. Kovalchuk, L. Wolf, and N. Dershowitz, "A simple and fast word spotting method," in *Proc. 14th International Conference on Frontiers in Handwriting Recognition (ICFHR), Crete, Greece*, September 2014, pp. 3–8.

[17] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "A novel word spotting method based on recurrent neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, 2012.

[18] B. Gatos and I. Pratikakis, "Segmentation-free word spotting in historical printed documents," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2009.

[19] J. A. Rodríguez-Serrano and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," in *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*, 2008.

[20] S. Espana-Boquera, M. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid HMM/ANN models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[21] J. Almazan, A. Gordo, A. Fornés, and E. Valveny, "Handwritten word spotting with corrected attributes," in *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2013.

[22] T. M. Rath and R. Manmatha, "Features for word spotting in historical manuscripts," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2003, pp. 218–222.

[23] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "HMM-based word spotting in handwritten documents using subword models," in *Proc. 20th International Conference on Pattern Recognition*, Aug 2010, pp. 3416–3419.

[24] L. Rothacker, M. Rusinol, and G. A. Fink, "Bag-of-features HMMs for segmentation-free word spotting in handwritten documents," in *Proc. 12th International Conference on Document Analysis and Recognition (ICDAR)*, Aug. 2013, pp. 1305–1309.

[25] N. Har'El and D. Kenigsberg, "Hspell – the free Hebrew spell checker and morphological analyzer," in *Israeli Seminar on Computational Linguistics*, December 2004.

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[27] Q. Liao, J. Z. Leibo, Y. Mroueh, and T. A. Poggio, "Can a biologically-plausible hierarchy effectively replace face detection, alignment, and recognition pipelines?" *Computing Research Repository (CoRR)*, vol. abs/1311.4082, 2013. [Online]. Available: http://arxiv.org/abs/1311.4082

[28] J. Han, *Data Mining: Concepts and Techniques*. San Francisco, CA: Morgan Kaufmann, 2005.

[29] A. Huang, "Similarity measures for text document clustering," Christchurch, New Zealand, 2008.