*Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pages 4717–4725
Marseille, 11–16 May 2020

# Time-Aware Word Embeddings for Three Lebanese News Archives

**Jad Doughman, Fatima K. Abu Salem, Shady Elbassuoni**
Computer Science Department
American University of Beirut
jad17@mail.aub.edu, fa21@aub.edu.lb, se58@aub.edu.lb

## Abstract

Word embeddings have proven to be an effective method for capturing semantic relations among distinct terms within a large corpus. In this paper, we present a set of word embeddings learnt from three large Lebanese news archives, which collectively consist of 609,386 scanned newspaper images and spanning a total of 151 years, ranging from 1933 till 2011. The diversified ideological nature of the news archives alongside the temporal variability of the embeddings offer a rare glimpse onto the variation of word representation across the left-right political spectrum. To train the word embeddings, Google's Tesseract 4.0 OCR engine was employed to transcribe the scanned news archives, and various archive-level as well as decade-level word embeddings were learnt. To evaluate the accuracy of the learnt word embeddings, a benchmark of analogy tasks was used. Finally, we demonstrate an interactive system that allows the end user to visualize for a given word of interest, the variation of the top-k closest words in the embedding space as a function of time and across news archives using an animated scatter plot.

**Keywords:** lebanese news archives, word embeddings, optical character recognition

## 1. Introduction

The public's perception is capable of shaping a nation's social and economic demeanour, thus making the source of that perception a vital matter. In the postmodern era, news media were restricted to local stations covering regional and international news, thus limiting the frequency and variety by which people obtained information. Citizens of under-developed countries relied predominantly on their primary local newspaper stations for the latest insight on political, economic, and cultural updates. Despite the vast amount of newspaper archives that has been generated throughout the years, historians have attempted to manually analyze this data for decades. The daunting process of transcribing such large archives propels both historians and computational linguists to try to find ways for automating the analysis of such a corpora through modern feature learning techniques in natural language processing (NLP), mainly word embeddings.

In this paper, we describe Arabic word embeddings trained using three large Lebanese news archives, namely: Assafir[1], Alhayat[2], and Annahar[3]. The diversified history of the three newspaper archives provides a more representative sample of ideologies from both the center-right and center-left spectrum. The three archives combined consist of 609,386 scanned newspaper images, which span a period of 151 years, ranging from 1933 till 2011. To transcribe our large corpus, an adequate OCR engine had to be selected. To facilitate this process, a ground truth consisting of a manual transcription of 5 newspaper images was generated and tested against various OCR engines and numerous configurations. Google's Tesseract 4.0 OCR engine, which returned the highest F-measure, was selected and used to transcribe our archives. In an effort to decrease the execution time, a multiprocessing script was written that ran multiple parallel instances of Tesseract 4.0. The OCRed archives were then used to learn the distributed representations of their words.

Word embedding models are a form of word representation that extends the human understanding of linguistics to a quantitative measure on a machine. Rational word embeddings are capable of capturing both the semantic and task-specific features of words. Hence, training time-aware word embedding models offers a rare glimpse into the manner in which word representations have been altered as a function of time. After transcribing our archives, various decade-level and archive-level Word2Vec models were trained (Mikolov et al., 2013b). To assess the quality of the trained word embeddings, a benchmark of analogy tasks is used (Elrazzaz et al., 2017). A word analogy question for a tuple consisting of two word pairs $(a, b)$ and $(c, d)$ can be formulated as follows: "$a$ to $b$ is like $c$ to ?".

Finally, we build an interactive system to visualize the movement of the top-k neighbors to an input word in the embedding space through an animated scatter plot. The results of the top-20 most similar words associated with the Lebanese Civil War were described in an effort to highlight the significance of temporal visualization and its ability to introduce relevant terms onto the mapped space across historical time periods. This would enable historians to seemingly analyze trends in word representation as a function of time and political spectra.

## 2. Flowchart

The flowchart shown in Figure 1 provides an overview of the relationship between the sections of this paper. The flowchart starts with using the *Original Images* for *OCR Engine Selection* section by manually generating the ground truth of five newspaper samples and testing them against various configurations of two open-source OCR engines: *Tesseract 4.0* and *Kraken.io*. With Tesseract 4.0 returning the higher F-measure value, various configurations of Tesseract 4.0 were tested including *Tesseract 4.0 with*

[1] http://assafir.com/
[2] http://www.alhayat.com/
[3] https://www.annahar.com/

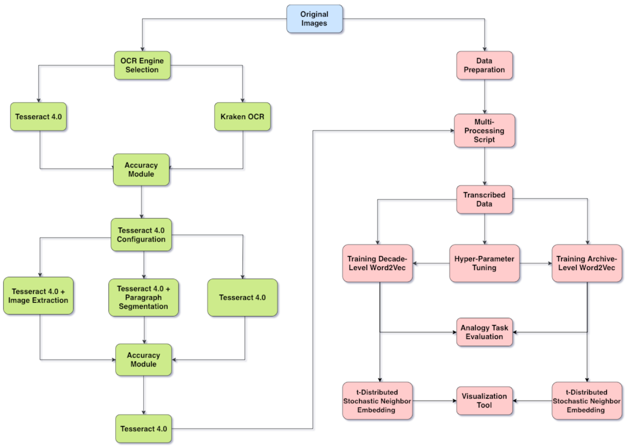
Original Images
OCR Engine Selection
Tesseract 4.0
Kraken OCR
Accuracy Module
Tesseract 4.0 Configuration
Tesseract 4.0 + Image Extraction
Tesseract 4.0 + Paragraph Segmentation
Tesseract 4.0
Accuracy Module
Tesseract 4.0
Data Preparation
Multi-Processing Script
Transcribed Data
Training Decade-Level Word2Vec
Hyper-Parameter Tuning
Training Archive-Level Word2Vec
Analogy Task Evaluation
t-Distributed Stochastic Neighbor Embedding
Visualization Tool
t-Distributed Stochastic Neighbor Embedding

Figure 1: Flowchart

*Image Extraction*, *Tesseract 4.0 with Paragraph Segmentation*, and plain *Tesseract 4.0*. Based on the F-measures of all the Tesseract configurations, standalone Tesseract 4.0 was selected and used in the *Multi-Processing Script* developed to efficiently transcribe our archives. The *Transcribed Data* was used to train decade-level and archive-level Word2Vec models using the tuned hyperparameters. The trained Word2Vec models were evaluated using the *Analogy Task Evaluation* on a benchmark of analogy tasks for the Arabic language. The resulting models are publically available here. *t-Distributed Stochastic Neighbor Embedding* was used to reduce the dimensional space of the models to be used in the *Visualization Tool*.

## 3. Literature Review

(Vasilopoulos et al., 2018) experimented with layout analysis designs to segment newspaper images into text and non-text block. They combined layout analysis and text localization in an attempt to improve OCR accuracy. (Pletschacher et al., 2015) developed a pipeline for the evaluation of OCR software on a dataset of historical newspaper images through the ground truth created. They examined specific types of errors in an effort to determine ways to improve image analysis and recognition.

(Clausner et al., 2013) presented a a sophisticated reading order representation scheme used by the system allowing the grouping of objects with ordered and/or unordered relations. This system was evaluated on state-of-the-art OCR and layout analysis systems. (Hadjar and Ingold, 2003) evaluated the ability of modern segmentation algorithms to deal with complex structured Arabic documents such as newspapers. They concluded that automatic layout analysis is not fully equipped to handle complex structured documents such as newspapers due to the variability of the layout.

(Hamilton et al., 2016) developed a robust methodology for quantifying semantic change by evaluating word embeddings (PPMI, SVD, word2vec) against known historical changes. They then used this methodology to reveal statistical laws of semantic evolution. Using six historical corpora spanning four languages and two centuries, they proposed two quantitative laws of semantic change: (i) the law of conformity – the rate of semantic change scales with an inverse power-law of word frequency; (ii) the law of innovation – independent of frequency, words that are more polysemous have higher rates of semantic change.

(Yao et al., 2017) developed a dynamic statistical model to learn time-aware word vector representation. They proposed a model that simultaneously learns time-aware embeddings and solves the resulting ”alignment problem”. Their model was trained on a crawled NYTimes dataset. Additionally, they developed multiple intuitive evaluation strategies of temporal word embeddings.

(Hämäläinen and Hengchen, 2019) presented an unsupervised learning technique for OCR post-correction. They developed a method of automatically extracting parallel data from their corpus. The parallel data extracted was used to train their sequence-to-sequence OCR post-correction model, which utilizes character-level neural machine translation.

## 4. News Archives Overview

### 4.1. Source and History

Our three newspaper archives were obtained from the American University of Beirut (AUB)'s University Libraries. Alongside scanned newspaper archives, the AUB Archives collections feature numerous noteworthy historical documents. The three primary preserved newspaper archives at AUB are Assafir, Alhayat, and Annahar. The Annahar newspaper provided nonconformists with a platform to express and relay their viewpoints during the years of the Syrian occupation of Lebanon. The paper can be thought of as centre-left, even though the views it has historically expressed spanned the entire political spectrum at times. Journalist Charles Glass argues that Annahar is “Lebanon's equivalent of The New York Times”[4]. According to Kamalipour and Mowlana's “Mass media in the Middle East : a comprehensive handbook”, Annahar had the highest circulation in Lebanon in the mid-1990s[5]. However, entering the 2000s, its circulation was at 45,000 copies, making it the second after Assafir's, at 50,000 copies. Assafir was founded in 1974 and represented Muslim interests, firmly promoting Arab nationalism, and was consistently pro-Syrian[6]. It has been cited to be the primary rival of Annahar in the region. The dynamic of having two dominant yet diverse mediums of data offer a discern onto the right and left centered ideologies in Lebanon. Alhayat newspaper was founded in 1946 as an independent and is regarded as being “more critical of the Saudi government than its rival.”, according to a 1997 article in The New York

[4] https://www.charlesglass.net/the-lord-of-no-mans-land-a-guided-tour-through-lebanons-ceaseless-war/

[5] https://searchworks.stanford.edu/view/2920917

[6] http://www.pressreference.com/Ky-Ma/Lebanon.html

Times[7]. That same article, also described Alhayat as a "decidedly Arab nationalist paper".

### 4.2. Assafir Archive

The Assafir archive consists of 185,147 scanned newspaper images (150 DPI). The total number of issues is 12,058, averaging around 16 pages per issue. The time frame for this archive ranges from years 1974 till 2011, accounting for a total of 37 years.

### 4.3. Alhayat Archive

The Alhayat archive, which is the smallest of the three, consists of 145,460 scanned newspaper images (150 DPI). The total number of issues is 11,325, averaging around 13 pages per issue. The time frame for this archive ranges from years 1950 till 1976 and 1988 till 2000, accounting for a total of 38 years.

### 4.4. Annahar Archive

The Annahar archive, which is largest of the three, consists of 278,779 scanned newspaper images (150 DPI). The total number of issues is 23,100, averaging around 12 pages per issue. The time frame for this archive ranges from years 1933-2009, accounting for a total of 76 years.

## 5. News Archives OCR

### 5.1. Optical Character Recognition Process

The Optical Character Recognition (OCR) operation incorporates image pre-processing, layout analysis, recognition, and sometimes, though rarely, an element of post-processing. Initially, pre-processing is applied, invoking imaging functions such as image rotation, de-skewing, and binarization, which is the process of converting a pixel image to a binary image using various thresholding algorithms. After processing the image, layout analysis is used to partition the digital image into multiple segments (sets of pixels, also known as super-pixels). The segmentation occurs at the level of paragraphs, lines, and subsequent words. Some OCR engines such as Tesseract split recognition into a two-pass process. During the first pass, each satisfactory word is used by an adaptive classifier as training data. Hence, the adaptive classifier gets a chance to learn how to recognize more accurately the text lower down the page. The adaptive classifier may sometimes learn something beneficial in the later stages of the page, hence a second pass is run over the page to fix any initially unrecognized words (Smith, 2007).

### 5.2. Layout Analysis Issue

The ambiguity around Arabic newspaper layout analysis can be attributed to the complex layout structure of Arabic newspapers, the variety of diacritic positioning, and overlapping text-lines. In the Arabic language, diacritic marks can have varying positions above or below the words. Due to the fact that some newspaper images consist of closely fitted text lines, the diacritic mark of the upper-line words would sometime overlap the characters of the lower-line words. The work in (Hadjar and Ingold, 2003) was based on experimenting with various layout analysis techniques, but concluded that for diacritic mark intersection, one faces "an ambiguity especially when diacritics of the first line and those of the second line are near each other or merged" (Hadjar and Ingold, 2003). The merged diacritic marks of two distinct words would hinder the OCR engine's accuracy due to its inability to parse the letters of each word separately.

### 5.3. OCR Engine Selection

We experimented with two open-source OCR engines. To evaluate the performance of each OCR engine, we generated ground truth consisting of five sample news articles and their full manual transcription as follows. We ran a random OCR engine on five randomly sampled newspaper articles. This resulted in a partially correct transcription of each newspaper image, which was then manually corrected to achieve five fully-transcribed samples. The five samples were then transcribed using Kraken.io[8], a turnkey OCR system forked from Ocropus, and Tesseract 4.0[9]. The pre-trained model used to test Tesseract 4.0 engine is *ara.traineddata*[10]. The pre-trained model used to test Kraken's OCR engine is *arabic_generalized.mlmodel*[11]. According to Kraken, the *arabic_generalized.mlmodel* is "the current best generalized model ... has also been trained with modern typefaces that are more aligned with the ones in current use than the older ones"[12]. As shown in Table 1, Tesseract outperformed Kraken, mainly due to its updated 4.0 engine with LSTM network and Leptonica's [13] advanced layout analysis.

| Engine | Precision | Recall | F1 score |
|---|---|---|---|
| Tesseract 4.0 | 0.753 | 0.868 | **0.805** |
| Kraken OCR | 0.216 | 0.319 | 0.257 |

Table 1: Word-level accuracy of Tesseract 4.0 and Kraken.io

### 5.4. Tesseract 4.0 Configurations

After opting for using Tesseract 4.0 as our OCR engine of choice, several image-processing configurations were tested for possible improvements in accuracy. The three main configurations were: (1) Running plain Tesseract 4.0 (2) Running Tesseract 4.0 with image extraction by manually extracting the images from the newspaper prior to

[7] https://www.nytimes.com/2005/02/06/weekinreview/spreading-the-word-whos-who-in-the-arab-media.html

[8] http://kraken.re

[9] https://github.com/tesseract-ocr/tesseract

[10] https://github.com/tesseract-ocr/tessdata/blob/master/ara.traineddata

[11] https://github.com/OpenITI/OCR_GS_Data/blob/master/ara/abhath/arabic_generalized.mlmodel

[12] https://github.com/mittagessen/kraken/issues/121#issuecomment-462284373

[13] http://www.leptonica.org

| Configuration | Precision | Recall | F1 Score |
|---|---|---|---|
| Tesseract 4.0 | 0.753 | 0.868 | **0.805** |
| Tess 4.0 + Img Ext | 0.779 | 0.777 | 0.778 |
| Tess 4.0 + Par Seg | 0.738 | 0.793 | 0.765 |

Table 2: Word-level accuracy of various Tesseract 4.0 configurations

OCRing ("Tess 4.0 + Img Ext" in Table 2 below) (3) Running Tesseract 4.0 with paragraph segmentation by manually segmenting paragraphs and running the OCR engine separately on each segment ("Tess 4.0 + Par Seg" in Table 2 below). This table shows that although processing the image prior to OCRing improved the precision, it hindered the recall value, leaving Tesseract 4.0 alone the primary choice with the highest F-measure value.

### 5.5. Tesseract 4.0 Multi-Processing

In an effort to decrease the execution time of the OCR process, a multi-processing script [14] was created using Python's multi-processing module, and executed on AUB's High Performance Computing (HPC) Arza cluster. The cluster consists of sixteen compute-nodes where each node in turn consists of 2 CPU sockets x 2.4 GHz Intel Sandy Bridge E5-2665 (8 physical cores per socket) and 64 GB of RAM (4GB per core). As such, we were able to spawn sixteen 16 worker threads on each compute-node, each one constantly processing OCR jobs. Additionally, four compute-nodes were invoked in parallel on the cluster, resulting in 64 concurrent instances of Tesseract. The expected execution time decreased to 10 days. We also experimented with compiling Tesseract 4.0 using gcc-8.1.0, which led to an additional 18% speedup.

## 6. Word Embedding Models

### 6.1. Training Word Embedding Models

After successfully transcribing all three archives, several word embedding models were trained on the transcribed data. The data initially got tokenized, normalized, and cleaned. The normalization process unifies the orthography of the first vowel letter in Arabic which comes in short and long spellings[15], while data cleaning removed punctuation marks and special characters. The tokenized words (forming sentences) were trained using a prediction-based embedding model (Word2Vec) (Mikolov et al., 2013b). To train the embeddings, Gensim library was used[16] . Choosing adequate values for Gensim's hyperparameters required thorough experimentation, which is discussed in Sec. 6.3.. To achieve the time-awareness aspect of the embeddings, a separate Word2Vec model was trained on a decade's worth of transcribed newspaper data. Additionally, an archive-level model was trained using the compilation of all the OCRed data of that archive.

### 6.2. Evaluating Word Embedding Models

To assess the quality of the trained word embeddings, a benchmark of analogy tasks for the Arabic language was used (Elrazzaz et al., 2017). The benchmark consisted of nine relations, each consisting of over a hundred word pairs. Given the benchmark, a test bank consisting of over 100,000 tuples was generated. Each tuple consisted of two word pairs $(a, b)$ and $(c, d)$ from the same relation. For each of the nine relations, a tuple was generated by combining two different word pairs from the same relation. Once tuples were generated, they were then used as word analogy questions to evaluate different word embeddings as defined by (Mikolov et al., 2013a). A word analogy question for a tuple consisting of two word pairs $(a, b)$ and $(c, d)$ can be formulated as follows: "$a$ to $b$ is like $c$ to ?". Each such question would then be answered by calculating a target vector $t = b - a + c$. We then calculated the cosine similarity between the target vector $t$ and the vector representation of each word $w$ in a given word embeddings $V$. Finally, we retrieved the most similar word $w$ to $t$, i.e., $argmax_{w \in V \& w \notin \{a,b,c\}} \frac{w \cdot t}{||w||||t||}$. If $w = d$ (i.e., the same word), then we assumed that the word embedding $V$ had answered the question correctly. Additionally, we extended the traditional word analogy task by taking into consideration whether the correct answer happend to be among the top-5 closest words in the embedding space to the target vector $t$, which allowed us to more leniently evaluate the embeddings. This is particularly important in the case of Arabic since many forms of the same word exist, usually with additional prefixes or suffixes, such as the equivalent of the article "the" or possessive pronouns such as "her", "his", or "their". To relax this and ensure that different forms of the same word will not result in a mismatch, we used the top-5 and top-10 words for evaluation rather than the top-1. Furthermore, the accuracy of each top-$k$ configuration was computed over OOV Penalty=True, which produces zero accuracy for 4-tuples with out-of-vocabulary words, and OOV Penalty=False, in which tuples with out-of-vocabulary words are skipped entirely and not used in the evaluation.

### 6.3. Hyperparameters

#### 6.3.1. Description

We begin with a description of each hyperparameter we had to configure. The *size* hyperparameter is the dimension of the word vectors. The *min_count* hyperparameter is the minimum number of occurrences for a word to be considered when training the model; words with occurrence less than this count would be ignored. The last hyperparameter is *sg*, which is either *0*, to train a CBOW model, which maximizes the probability of the target word by looking at the context, or *1*, to train a skip-gram model, which when given a word attempts to predict its context.

#### 6.3.2. Hyperparameter Tuning

Due to the unique circumstance of having to use imperfect OCRed text to train word embedding models, using generic values for certain hyperparameters was not a viable option. For example, the hyperparameter *min_count* is generically set to 5. However, due to various text spelling

[14] `https://github.com/jaddoughman/Multiprocessing-Tesseract-4.0`

[15] For a reader of Arabic, this includes alifs, hamzas, and yas/alif maqsuras

[16] `https://github.com/RaRe-Technologies/gensim`

| Archive | OOV Penalty | False | | | True | | |
|---|---|---|---|---|---|---|---|
| | **Model/Top-K** | **Top-1** | **Top-5** | **Top-10** | **Top-1** | **Top-5** | **Top-10** |
| **Assafir** | **1974_1983** | 11.49% | 22.36% | 27.28% | 2.43% | 4.73% | 5.77% |
| | **1984_1993** | 11.20% | 22.36% | 27.24% | 2.71% | 5.41% | 6.59% |
| | **1994_2003** | 10.92% | 20.82% | 25.44% | 3.12% | 5.95% | 7.28% |
| | **2004_2011** | 11.01% | 21.67% | 26.82% | 2.55% | 5.02% | 6.21% |
| **Alhayat** | **1950_1959** | 8.39% | 17.23% | 21.33% | 0.81% | 1.67% | 2.07% |
| | **1960_1969** | 10.16% | 19.98% | 24.44% | 1.46% | 2.87% | 3.51% |
| | **1970_1976** | 10.32% | 21.38% | 26.27% | 0.93% | 1.92% | 2.36% |
| | **1988_1989** | 20.57% | 36.88% | 42.50% | 1.35% | 2.43% | 2.80% |
| | **1990_2000** | 12.57% | 25.07% | 30.36% | 3.866% | 7.70% | 9.33% |
| **Annahar** | **1933_1942** | 5.49% | 11.96% | 15.38% | 0.26% | 0.56% | 0.72% |
| | **1943_1952** | 5.35% | 12.88% | 16.34% | 0.18% | 0.44% | 0.56% |
| | **1953_1962** | 8.09% | 16.68% | 20.86% | 0.97% | 2.00% | 2.49% |
| | **1963_1972** | 8.71% | 18.08% | 22.20% | 1.40% | 2.92% | 3.59% |
| | **1973_1982** | 8.92% | 18.99% | 23.76% | 1.90% | 4.04% | 5.05% |
| | **1983_1992** | 9.40% | 21.38% | 26.41% | 2.30% | 5.24% | 6.47% |
| | **1993_2002** | 11.56% | 21.18% | 25.55% | 3.08% | 5.65% | 6.82% |
| | **2003_2009** | 10.74% | 21.06% | 25.61% | 2.70% | 5.30% | 6.45% |

Table 3: Evaluation of decade-level models trained on each transcribed news archive

errors caused by the OCR engine's inability to segment the overlapping text-lines correctly, a diminutive value for *min_count* would not do well. To choose the best hyperparameter values, we sampled a 70,000 transcribed batch and trained 12 CBOW and skip-gram models, with *min_count* values = [10,100,300,500] and *size* values = [200,250,300]. The CBOW models outperformed skip-gram models due to our data having high frequency words. This is a typical scenario for the two models: skip-gram tends to perform better on rather a small amount of the training data and to represent well even rare words or phrases, as opposed to CBOW that is not only several times faster to train than the skip-gram, but obtains slightly better accuracy for the frequent words[17]. We opted on using min_count=100 and size=300 as generic values for these hyper-parameters since they returned the highest accuracy.

| Archive | Assafir | | | | | |
|---|---|---|---|---|---|---|
| **OOV Penalty** | **False** | | | **True** | | |
| **Rel / Top-K** | **Top-1** | **Top-5** | **Top-10** | **Top-1** | **Top-5** | **Top-10** |
| **Capitals** | 29.8% | 53.2% | 61.1% | 11.1% | 19.9% | 22.8% |
| **Currency** | 0.1% | 1.6% | 4.0% | 0.1% | 0.5% | 1.2% |
| **Man-Woman** | 3.2% | 10.3% | 15.6% | 1.0% | 3.4% | 5.1% |
| **Nationality** | 33.8% | 49.1% | 53.6% | 12.1% | 17.6% | 19.2% |
| **Plurals** | 7.3% | 23.1% | 30.8% | 3.5% | 11.1% | 14.8% |
| **Comparative** | 10.2% | 19.0% | 24.0% | 8.6% | 16.0% | 20.3% |
| **Opposite** | 3.2% | 10.4% | 14.6% | 2.2% | 7.1% | 10.0% |
| **Pairs** | 5.3% | 14.2% | 20.0% | 0.2% | 0.5% | 0.8% |
| **Past Tense** | 4.3% | 12.0% | 16.2% | 2.8% | 7.8% | 10.6% |
| **Total** | 9.8% | 19.9% | 24.9% | 4.3% | 8.7% | 10.9% |

Table 4: Evaluation of Assafir archive-level model across various relations

| Archive | Alhayat | | | | | |
|---|---|---|---|---|---|---|
| **OOV Penalty** | **False** | | | **True** | | |
| **Rel / Top-K** | **Top-1** | **Top-5** | **Top-10** | **Top-1** | **Top-5** | **Top-10** |
| **Capitals** | 34.9% | 59.6% | 67.9% | 11.7% | 20.0% | 22.7% |
| **Currency** | 0.8% | 3.1% | 6.0% | 0.3% | 1.0% | 2.0% |
| **Man - Woman** | 2.3% | 9.3% | 14.0% | 0.7% | 2.8% | 4.3% |
| **Nationality** | 34.2% | 47.7% | 51.0% | 12.2% | 17.0% | 18.2% |
| **Plurals** | 10.9% | 29.3% | 37.8% | 4.5% | 12.3% | 15.8% |
| **Comparative** | 12.7% | 23.6% | 28.8% | 10.0% | 18.7% | 22.8% |
| **Opposite** | 3.1% | 12.5% | 16.7% | 2.0% | 7.8% | 10.4% |
| **Pairs** | 14.5% | 27.3% | 39.1% | 0.2% | 0.3% | 0.4% |
| **Past Tense** | 4.5% | 12.0% | 16.3% | 2.7% | 7.2% | 9.7% |
| **Total** | 11.1% | 22.1% | 27.1% | 4.6% | 9.1% | 11.1% |

Table 5: Evaluation of Alhayat archive-level model across various relations

| Archive | Annahar | | | | | |
|---|---|---|---|---|---|---|
| **OOV Penalty** | **False** | | | **True** | | |
| **Rel / Top-K** | **Top-1** | **Top-5** | **Top-10** | **Top-1** | **Top-5** | **Top-10** |
| **Capitals** | 30.7% | 48.9% | 56.1% | 10.9% | 17.3% | 19.9% |
| **Currency** | 0.1% | 1.6% | 3.2% | 0.1% | 0.6% | 1.1% |
| **Man - Woman** | 3.4% | 10.1% | 13.5% | 1.2% | 3.6% | 4.9% |
| **Nationality** | 32.3% | 48.2% | 52.1% | 10.8% | 16.1% | 17.4% |
| **Plurals** | 6.7% | 18.8% | 26.2% | 3.6% | 10.2% | 14.2% |
| **Comparative** | 10.4% | 21.2% | 26.1% | 9.0% | 18.4% | 22.6% |
| **Opposite** | 2.4% | 10.2% | 14.4% | 1.9% | 7.7% | 11.0% |
| **Pairs** | 4.4% | 14.6% | 18.6% | 0.2% | 0.7% | 1.0% |
| **Past Tense** | 4.7% | 11.5% | 15.3% | 3.2% | 7.8% | 10.4% |
| **Total** | 8.9% | 18.1% | 22.5% | 4.2% | 8.5% | 10.6% |

Table 6: Evaluation of Annahar archive-level model across various relations

## 6.4. Results

The evaluation of the decade-level word embedding models trained on each of the three large archives is shown in

[17] https://groups.google.com/forum/#!searchin/word2vec-toolkit/c-bow/word2vec-toolkit/NLvYXU99cAM/E51d8LcDx1AJ

Table 3. A total accuracy, averaged across various relations, is displayed for each decade-level model. The results include the top-1, top-5, top-10 accuracy computed over OOV Penalty=True and OOV Penalty=False. The total accuracy for each configuration shown was averaged across the following relations: Capital Cities, Currency, Man-Women, Nationality, Plurals, Comparative, Opposite, Pairs, and Past Tense. The evaluation of the archive-level models trained on each of the Assafir, Alhayat, and Annahar archives are shown in Table 4, 5, and 6 respectively. Each table includes a final aggregate report with a top-1, top-5, top-10 accuracy result for each relation when OOV Penalty=True and OOV Penalty=False.

# 7. Visualization Tool

## 7.1. T-distributed Stochastic Neighbor Embedding

T-SNE is a machine learning algorithm for nonlinear dimensionality reduction. The fundamental concept is to reduce the dimensional space, but by maintaining the relative pairwise distance between points. It helps visualize high-dimensional data by giving each datapoint a location in a two or three-dimensional map (van der Maaten and Hinton, 2008). The hyperparameters for t-SNE are:

- **Number of Components**: the dimension of the output space.
- **Perplexity**: a measure of the effective number of neighbors. Typically takes on values ranging from 5 to 50 (van der Maaten and Hinton, 2008).
- **Type of Initialization**: Principal component analysis (PCA) is selected since it is usually more globally stable than random initialization.

## 7.2. Post-Processing Using Levenshtein Distance

Due to the imperfect OCRed data used to train the word embedding models, a significant portion of the model's vocabulary would contain distinct entries for misspelled variants of a single word. A misspelled version of a given word will always appear in the same context as the original correct spelling of that word. For example: (i) *Yesterday was a really nice day* and (ii) *Yesterday was a really nife day*. This can be problematic since the Continuous Bag of Words (CBOW) model maximizes the probability of the target word by looking at the context. Thus, attempting to retrieve the top-k closest vectors by computing the cosine similarity between a simple mean of the projection weight vector of a given word and the vectors for each word in the model would result in a variant of the given word or its misspelled OCR version. An example of this issue is shown in Figure 2 under *Before Post-Processing*. Attempting to find the top-5 most similar words to افغانستان [18] would return five misspelled variants of that word. All five variants are within 1-3 single-character edits from the given word.

To overcome this matter in the visualization tool, whenever the end user inputs a word, the tool iterates over the most similar vectors of that given word and filters out all the results whose Levenshtein distance to the given word is less than or equal to two (the Levenshtein distance is the minimum number of single-character edits required to change one word into the other) given that the length of inputted word is greater than three characters. This removes all the results which are the stem/root or misspelled version of that given word, leaving us with true representative vectors. An example of applying post-processing using Levenshtein distance would result in the words shown in Figure 2 under *After Post-Processing*. Hence, finding the top-5 most similar words to افغانستان would now result in باكستان, العراق, اميركا, الصومال, and ليبيا[19], which are the actual results we are after.

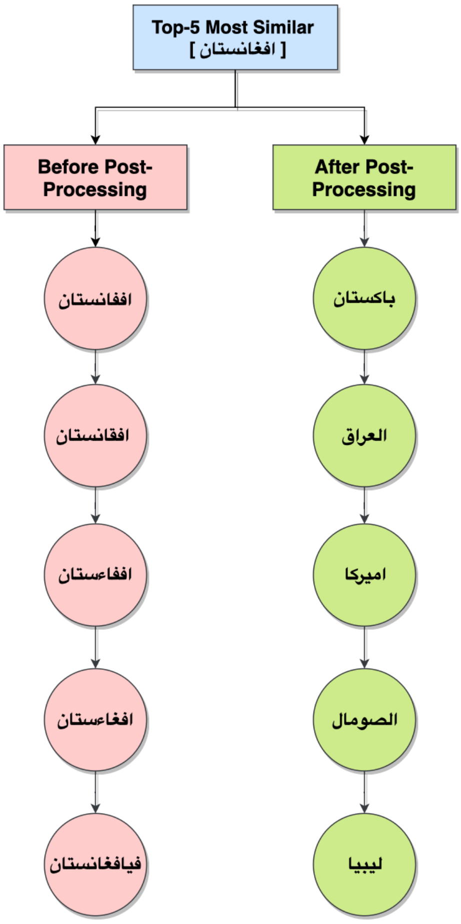

Figure 2: Top-5 most similar words before and after post-processing

## 7.3. Temporal Visualization

The interactive system allows for a temporal visualization of the embeddings by mapping the top-k most similar word vectors across the given time-frame onto a two or three-dimensional space. To accomplish this task, t-SNE needs to be applied on every decade-level model and the

[18] Translates to Afghanistan in English

[19] Translate respectively to Pakistan, Iraq, America, Somalia, and Libya in English

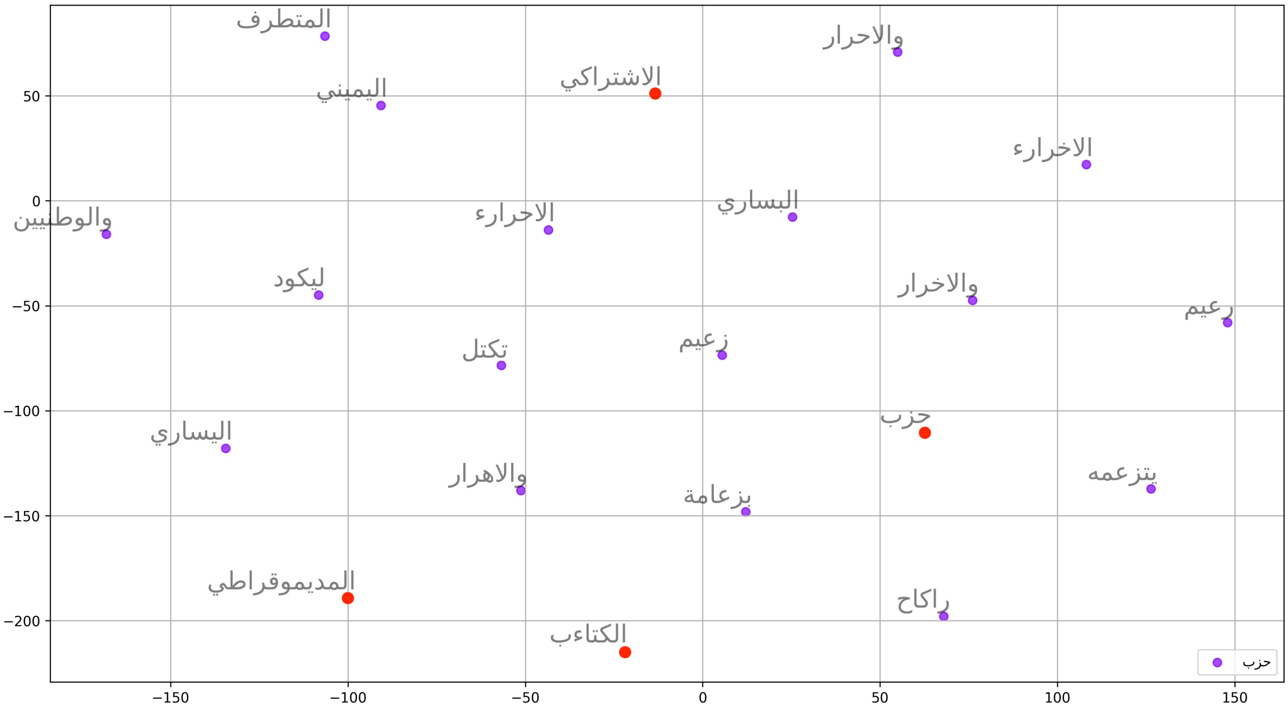

Figure 3: Top-20 most similar words to حزب in the Annahar 1973-1982 decade-level model

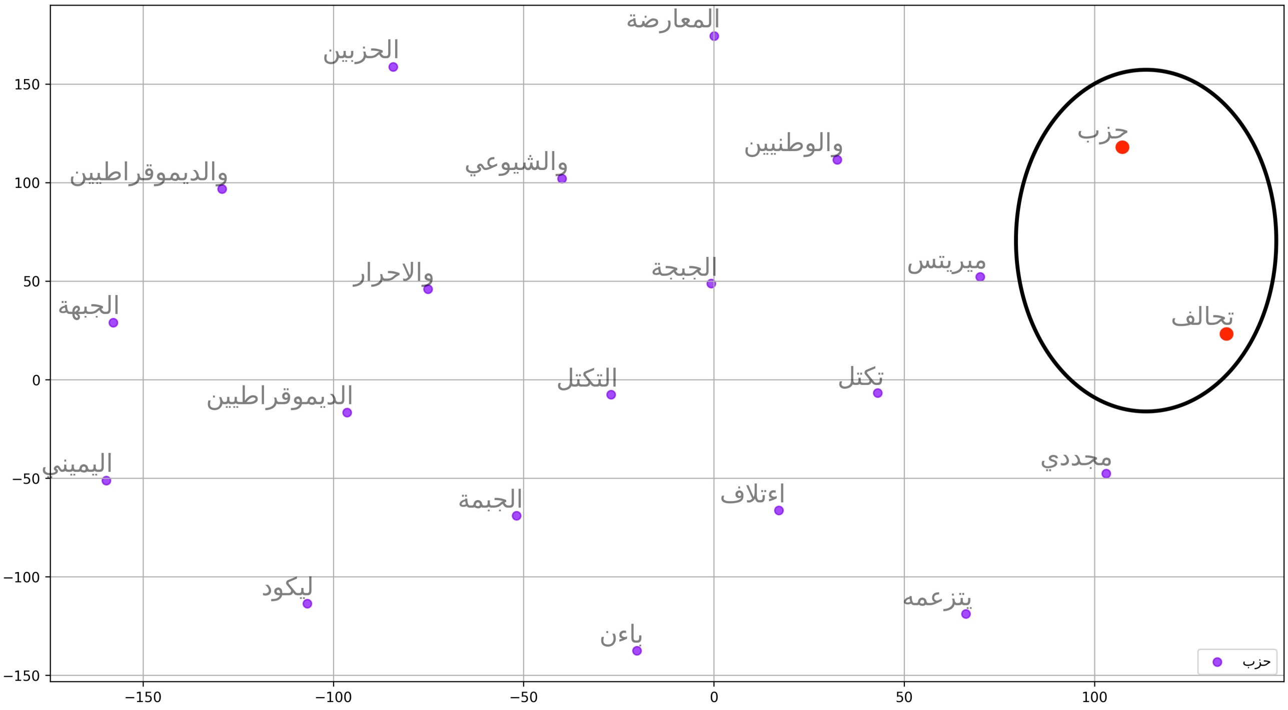

Figure 4: Top-20 most similar words to حزب in the Annahar 1983-1992 decade-level model

mapping of the top-k most similar word vectors would be employed on each model in ascending time-frame order, essentially forming a time-based animation. The end user would be able to visualize the movements of the word vectors as well as the introduction of new word vectors onto the top-k most-similar mapped space. This can be especially informative for historians attempting to seemingly understand the variation of word representation across time-frames and diverse news archives.

The introduction of certain word vectors during the animation can be aligned with historical/political events that occurred in that decade. The multifaceted Lebanese Civil War, which lasted from 1975 to 1990, included a

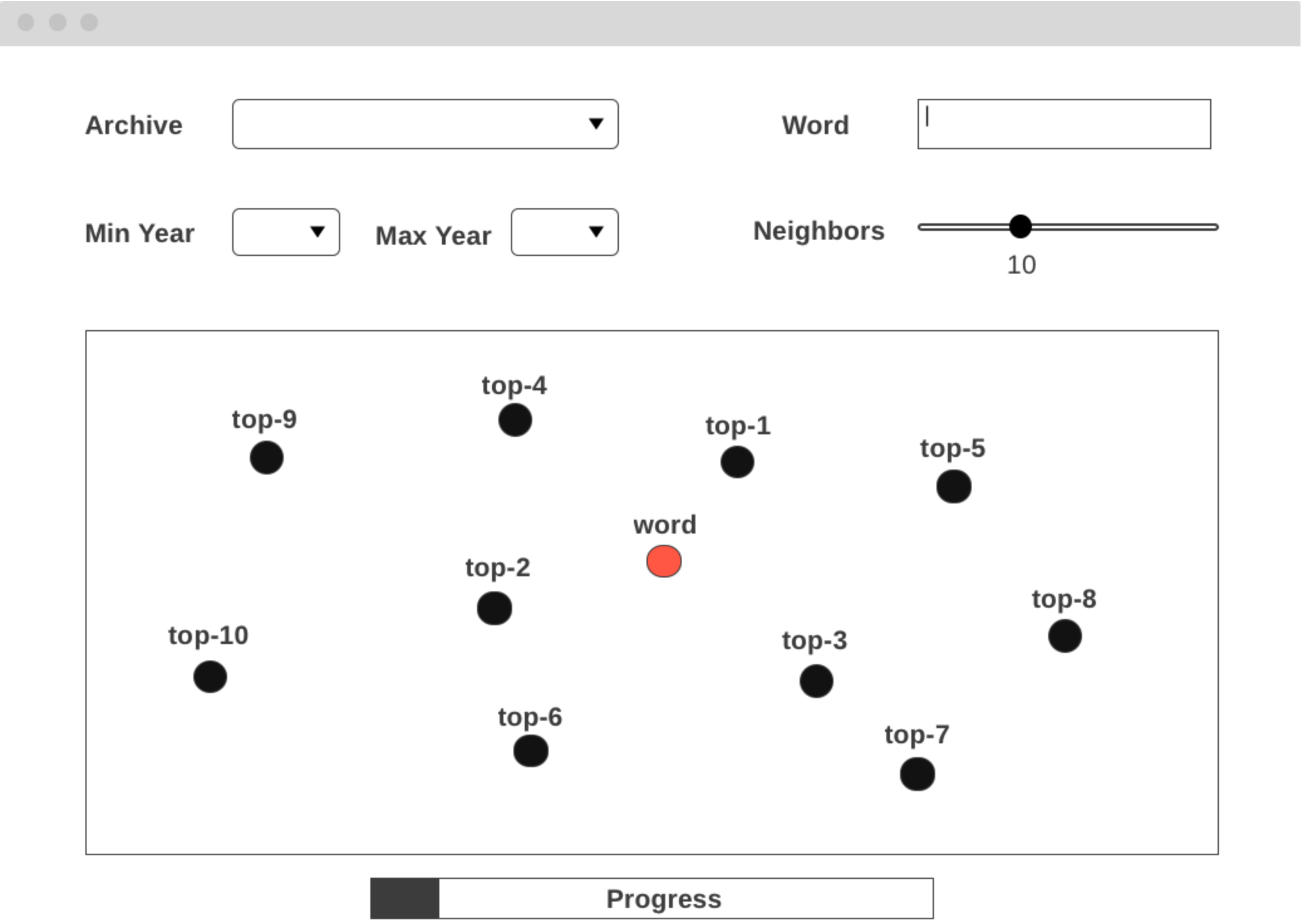

Figure 5: Visualization tool screen mock-up

wide range of belligerent political parties. Taking the decade-level Annahar model trained on data from 1973 to 1982, the top-20 most similar words to حزب[20] shown in Figure 3 results in the names of several political parties involved in the war, including the Progressive Socialist Party, Lebanese Democratic Party, and Kataeb Party. The Taif Agreement was reached during 1989 to provide the basis for the ending of the civil war and the return to political normalcy in Lebanon. Taking the decade-level Annahar model trained on data from 1983 to 1992 shown in Figure 4, the top-20 most similar words to حزب now introduces the word vector تحالف [21]. The newly introduced word in the 1983-1992 model perfectly aligns with the Taif agreement, a pan-Arab accord that brought all warring Lebanese parties together and signaled the end of the 15-year civil war.

### 7.4. Screen Mock-up

The screen mock-up shown in Figure 5 illustrates the user interface of the visualization tool. The goal is to visualize the movement of the top-k neighbors (most similar words) to the given word through an animated scatter plot. This would enable historians to seemingly analyze trends in word representation as a function of time. To generate the animated scatter plot, the tool takes as input:

- **Archive**: this comprises of a drop-down menu of three options (Assafir, Alhayat, Annahar).
- **Time-frame**: the *Min Year* and *Max Year* values
- **Word**: the word for which the end user wants to find the top-k most similar word vectors.
- **Number of Neighbors**: the number of top-k neighbors to be retrieved.

## 8. Conclusion

In this paper, we described Arabic word embeddings trained using three large Lebanese news archives, namely: Assafir, Alhayat, and Annahar. The three archives combined consist of 609,386 scanned newspaper images, which span a period of 151 years, ranging from 1933 till 2011. To transcribe our large corpus, Google's Tesseract 4.0 OCR engine was used. The OCRed archives were then used to train various decade-level and archive-level Word2Vec models. To assess the quality of the trained word embeddings, a benchmark of analogy tasks was used. Finally, we demonstrated an interactive system that allows the end user to visualize the movement of the top-k neighbors to an input word in the embedding space through an animated scatter plot.

[20] Translates to "political party" in English

[21] Translates to "alliance" or "pact" in English

## 9. Bibliographical References

Clausner, C., Pletschacher, S., and Antonacopoulos, A. (2013). The significance of reading order in document recognition and its evaluation. 08.

Elrazzaz, M., Elbassuoni, S., Shaban, K., and Helwe, C. (2017). Methodical evaluation of Arabic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 454–458, Vancouver, Canada, July. Association for Computational Linguistics.

Hadjar, K. and Ingold, R. (2003). Arabic newspaper page segmentation. pages 895–899, 01.

Hämäläinen, M. and Hengchen, S. (2019). From the paft to the fiiture: a fully automatic nmt and word embeddings method for ocr post-correction.

Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016). Diachronic word embeddings reveal statistical laws of semantic change. *CoRR*, abs/1605.09096.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, et al., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Pletschacher, S., Clausner, C., and Antonacopoulos, A. (2015). Europeana newspapers ocr workflow evaluation. pages 39–46, 08.

Smith, R. (2007). An overview of the tesseract ocr engine. volume 2, pages 629 – 633, 10.

van der Maaten, L. and Hinton, G. (2008). Viualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 11.

Vasilopoulos, N., Wasfi, Y., and Kavallieratou, E., (2018). *Automatic Text Extraction from Arabic Newspapers*, pages 505–510. 06.

Yao, Z., Sun, Y., Ding, W., Rao, N., and Xiong, H. (2017). Discovery of evolving semantics through dynamic word embedding learning. *CoRR*, abs/1703.00607.