# A Weighted Combination of Semantic and Syntactic Word Image Representations

Oliver Tüselmann[0000−0002−8892−3306], Kai Brandenbusch[0000−0001−5258−7454], Miao Chen[0000−0002−6403−1899], and Gernot A. Fink[0000−0002−7446−7813]

Department of Computer Science, TU Dortmund University,
44227 Dortmund, Germany
{firstname.lastname}@cs.tu-dortmund.de

**Abstract.** In contrast to traditional keyword spotting, semantic word spotting allows users to search not only for word images with the same transcription as the keyword, but also for concepts which are latent or hidden inside a query. However, it has been shown that mapping word images to semantic representations proves to be a difficult task. As semantic embeddings do not consider syntactic similarity, it is common to find search results with highly ranked semantically similar word images, while words with the same transcription as the search query appear in lower ranks. To counteract this problem, a combination of semantic and syntactic representations usually provides a good trade-off w.r.t. semantic and syntactic metrics. In this work, we present methods for realizing a weighted combination of semantic and syntactic information. This allows users to focus more on semantic or syntactic aspects and thus provides new insights to their document collections. Thereby, our proposed methods are not limited to the use of word spotting, but also aim to address the optimization of recognition-free NLP downstream tasks.

## 1 Introduction

In recent years, great progress has been made in the field of handwriting recognition [18]. However, the transformation of handwritten document images into digital text remains a difficult task, especially for small as well as historical datasets [17]. For this reason, there is a sustained interest in and use of recognition-free word spotting approaches [7]. The goal of word spotting is to retrieve word images from a collection of document images that are similar w.r.t. a given query. Traditional approaches define similarity based on visual properties and avoid an explicit recognition step. This leads to search results in which word images with syntactically close transcriptions to the query are highly ranked. Thereby, two words are syntactically similar if their string edit distance is small. Over the last few years, deep learning based approaches achieved remarkable results on most benchmark datasets in this domain [13,21,24].

An extension of this task is semantic word spotting [14,22,24], where semantic information is taken into account during retrieval. This allows users to not only search for syntactically similar words, but also for concepts which are latent or hidden inside a query. A concept could be a similar meaning (e.g. great and huge) or a categorical relationship (e.g. animal and cat). Searching for semantically similar occurrences of words in a given document collection offers users a new way to efficiently explore their collections. However, from a technical perspective, such a search poses major challenges [14,22]. This is mainly due to the high variability of handwriting, which often leads to an incorrect prediction of semantic information [14,22,24]. Another limiting factor is the challenging prediction of semantic information for words that have not been seen during training [14,22]. To overcome these limitations, Krishnan et al. have recently shown that a stacking of semantic and syntactic representations can yield promising results in terms of both semantic and syntactic evaluation measures [14].

However, a simple stacking of these embeddings often leads to a semantic or syntactic bias caused by the different characteristics of the representations (e.g. dimensions and dynamic ranges). In this work, we show how syntactic and semantic representations can be combined with equal importance. Thereby, we present methods for combining both semantic and syntactic information into a single representation by using a weighted combination. The adjustment of the weighting parameter is task-dependent and enables users to prioritize semantic or syntactic aspects and thus obtain new insights into their document collections.

The remainder of this paper is organized as follows. Section 2 introduces the basics and related work in the field of word spotting and word embeddings. In section 3, we present the approaches used for predicting semantic and syntactic embeddings from word images and how to combine them appropriately. We then evaluate the weighted combinations quantitatively as well as qualitatively on three handwriting datasets in section 4. Finally, we summarize the results in section 5.

## 2 Related Work

In this section, we provide an overview on traditional as well as semantic word spotting. We further introduce an outline of syntactic and semantic word representations used in the context of word spotting.

### 2.1 Traditional Word Spotting

Word spotting is a retrieval-based method for determining regions in a document image that are similar w.r.t. a given query. The approaches avoid an explicit text recognition and use visual features to determine the similarity between a word image and a query. The similarities are finally used to determine a retrieval list of the most similar word images w.r.t. a given query.

Word spotting approaches can be divided into multiple categories. There exists a variety of different query types with Query-by-Example (QbE) and Query-by-String (QbS) being the most prominent ones. In QbE applications, the query is a word image whereas in QbS it is a textual string representation. Furthermore, word spotting can be divided into segmentation-free (i.e. entire document images are used without any segmentation) and segmentation-based (i.e. a word level segmentation is required) approaches. There is a wide range of methods in this area covering Bag-of-Feature representations, sequence models, Support Vector Machines and Neural Networks [7]. Recently, approaches based on Convolutional Neural Networks (CNNs) have achieved remarkable results for most benchmark datasets in this area [13,21]. For a more detailed overview of word spotting, see [7].

## 2.2 Semantic Word Spotting

Semantic word spotting realizes a semantic word image retrieval and can be seen as an extension of the traditional word spotting approach. The aim of this task is to retrieve all word images with the same transcription as the query, followed by semantically similar ones. Even if there are multiple ways to achieve a semantic word image retrieval, we only consider approaches that predict semantic information directly from a word image without transcribing it.

The first approaches of semantic word spotting rely on ontology-based knowledge [8,11] and are thus limited to a small set of human labeled semantic relationships. To overcome this limitation, Wilkinson et al. [24] use a two-stage CNN-based approach to map word images into a textually pre-trained semantic space. Tueselmann et al. [22] show that the semantic space used in [24] encodes only few semantic relations and is close to a syntactic embedding in many parts. They suggest to use a pre-trained FastText [3] embedding and present an optimized two-stage architecture for mapping word images into a semantic space. Recently, Krishnan et al. [14] proposed an end-to-end approach for mapping word images into a semantic space and explored word normalization methods from the Natural Language Processing (NLP) domain, such as Lemmatization and Stemming. Furthermore, they showed that a stacking of semantic and syntactic representations is able to tackle major problems of just using semantic word image embeddings.

## 2.3 Word Embeddings

Word embeddings convert strings into vector representations. They are often used in word spotting systems to enable a comparison between word images and strings [2,21,24]. The Pyramidal Histogram of Characters (PHOC) embedding is the most commonly used embedding in the field of word spotting [2,13,21]. A PHOC is a binary pyramidal representation of a character string and is used to represent visual attributes of a given word image. Besides PHOC, the Discrete Cosine Transform of Words (DCToW) [24] and the Deep Embedding approach by Krishnan et al. [13] provide state-of-the-art results for many word spotting benchmark datasets.

For realizing a semantic word spotting approach, it is necessary to encode semantic relationships between word images. There are several approaches from the NLP domain that provide this kind of information for textual inputs [3,5,19,20]. The approaches can be divided into context-based and static word embeddings.

Context-based methods [5,20] calculate a word representation for a given word based on the context in which it appears. Whereas, static methods [3,19] always output the same embedding for a word, regardless of the context in which it occurs. Since words in different contexts often have different meanings, context-based models are preferable in most applications [6]. However, if there is no context information as in segmentation-based word spotting, static embeddings are preferable.

The combination of embeddings has already been successfully applied in the NLP domain [1,4,23]. The motivation for this combination is that different embeddings encode different semantic aspects and thereby, their combination leads to performance gains in many tasks [1,23]. In contrast to our work, the concatenated embeddings just serve as input to neural architectures and the equal importance of both embeddings is ignored.

## 3 Method

In this section, we present methods for realizing a weighted combination of semantic and syntactic word image representations. In section 3.1, we introduce the representations used and show how they are predicted from word images using CNNs. Then, we present two approaches for a weighted combination in section 3.2. Finally, we motivate in section 3.3 that a normalization of semantic and syntactic embeddings is necessary before stacking them, due to their different properties in dimensionality and ranges.

### 3.1 Word Image Representation

For realizing a weighted combination of semantic and syntactic information, suitable word image representations are needed. Similar to previous works in the semantic word spotting domain [14,22], we use FastText as the semantic representation. Due to the state-of-the-art performance in traditional word spotting and the ability to adjust the dimensionality of the embedding size, we use the HWNetv2 [13] for obtaining our syntactic representation.

We use the proposed networks from Krishnan et al. [14] for mapping word images to word representations. Thereby, the HWNetv2 [13] is used for predicting syntactic and a modified ResNet (Attribute-ResNet) for semantic features. The Attribute-ResNet uses a ResNet34 architecture [10] for feature extraction, whereby the global average pooling layer at the end of the network is replaced with a Temporal Pyramid Pooling (TPP) layer. The output of the TPP layer is transferred into a 3-layer Fully-Connected Network (FCN). This FCN has as many neurons in the last layer as there are dimensions in the word representation to be predicted (e.g. FastText=300). Except for the final layer, the ReLU activation function is applied to the output of all layers in the network.

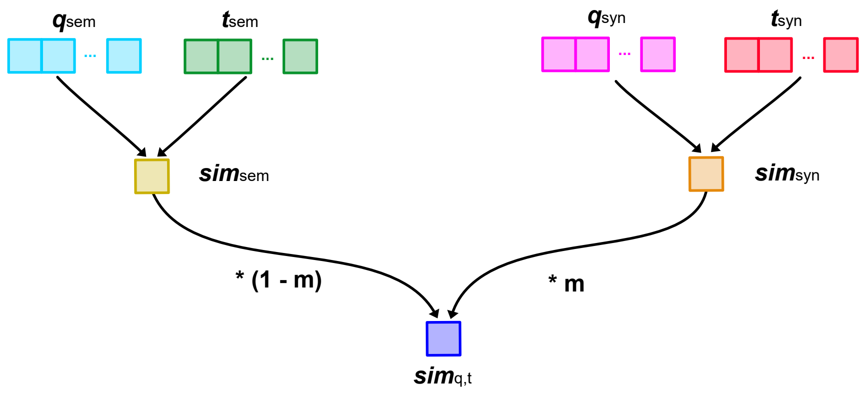


Fig. 1: An overview of the indirect weighting approach.

## 3.2 Weighted Combination Approaches

We propose two approaches for a weighted combination of semantic and syntactic embeddings. Both approaches receive a query, a word image from the test set, and a weighting factor $m \in [0, 1]$ as input and output a similarity score. In the following, the proposed methods are described in more detail.

We obtain the semantic ($q_{sem}$) and syntactic ($q_{syn}$) representations for a given query $q$ as already shown in the previous section. The query can be either a word image or a text string. Similarly, for an element of the test set $t$, the representations are denoted by $t_{syn}$ and $t_{sem}$. As a metric we employ the cosine similarity which is commonly used in retrieval tasks.

**Indirect Approach** The indirect approach follows an intuitive realization of a weighted combination. After computing the semantic and syntactic similarities separately, the weighted similarity score is obtained by using a linear interpolation (see Fig. 1). Technically, the cosine similarity $sim_{syn}$ of $q_{syn}$ and $t_{syn}$, as well as $sim_{sem}$ of $q_{sem}$ and $t_{sem}$ are computed respectively. Finally, $sim_{syn}$ and $sim_{sem}$ are weighted by a parameter $m$ and $(1-m)$ respectively, resulting in its new similarity score $sim_{q,t}$ (see Eq. 1).

$$\begin{aligned} sim_{syn} &= similarity(q_{syn}, t_{syn}) \\ sim_{sem} &= similarity(q_{sem}, t_{sem}) \\ sim_{q,t} &= m \cdot sim_{syn} + (1-m) \cdot sim_{sem} \quad (0 <= m <= 1) \end{aligned} \tag{1}$$

**Direct Approach** In contrast to the indirect approach, the semantic and syntactic vectors for both the query and the test image are first weighted element-wise. Afterwards, these weighted representations are concatenated and finally,

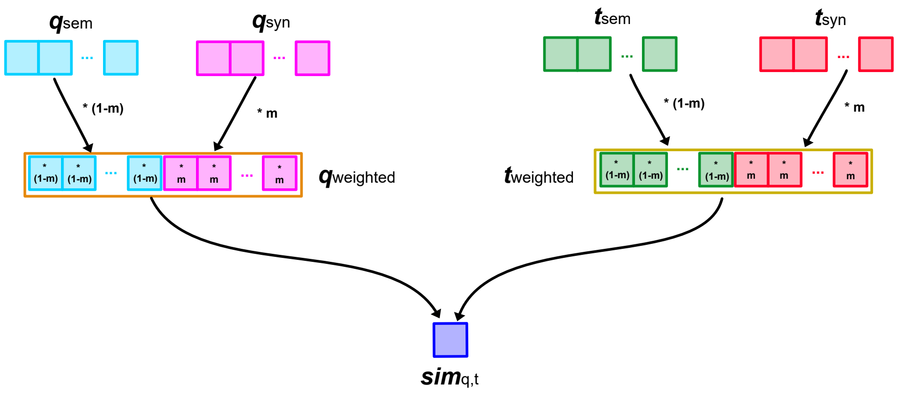


Fig. 2: An overview of the direct weighting approach.

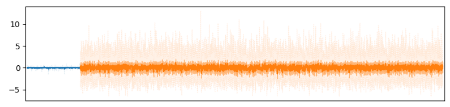
the similarity between the concatenated representations of the query and test element is calculated (see Fig. 2). A fundamental benefit of this approach is that the concatenated representations can be used not only for obtaining similarity scores, but also for recognition-free downstream tasks in the NLP domain. Technically, each element of the syntactic and semantic embeddings is multiplied with $m$ and $(1 - m)$ respectively. The weighted syntactic and semantic embeddings of $q$ and $t$ are then stacked, resulting in new representations $q_{weighted}$ and $t_{weighted}$. Finally, the cosine similarity of these two embeddings provides their new similarity score $sim_{q,t}$ (see equation 2).

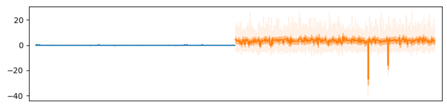
$$\begin{aligned} q_{weighted} &= concat((1-m)\cdot q_{sem}, m\cdot q_{syn}) \\ t_{weighted} &= concat((1-m)\cdot t_{sem}, m\cdot t_{syn}) \quad (0 <= m <= 1) \\ sim_{q,t} &= similarity(q_{weighted}, t_{weighted}) \end{aligned} \tag{2}$$

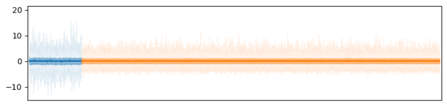
## 3.3 Normalization

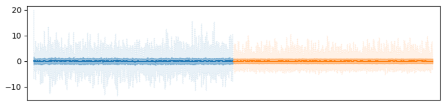
For the indirect combination of semantic and syntactic embeddings we can compute the distances in the original embedding spaces. For our approach of stacking the embeddings, however, we have to take the differences of the embedding spaces into consideration. First, the dimensionality of FastText (300-D) differs from the dimensionality of the HWNet embedding (2048-D). Second, the dynamic range of the features in the embeddings differ as shown in Fig. 3a.

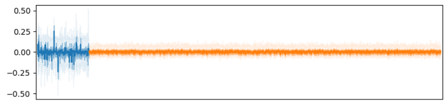
In order to mitigate these problems, we apply different normalization methods to the embeddings before the concatenation. The effects of the different normalization approaches to the statistics of the concatenated embeddings can be observed in Fig. 3. Following the approach of Krishnan et al. [14], we normalize the embeddings to have zero mean and unit variance (standardization, see Figs. 3c and 3d). In contrast to their work, we standardize both embeddings

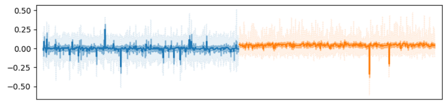

(a) FT & HW2048 (base)

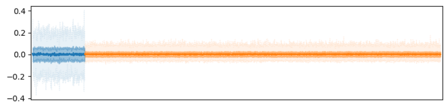

(b) FT & HW300 (base)


(c) FT & HW2048 (std.)


(d) FT & HW300 (std.)


(e) FT & HW2048 (l2)


(f) FT & HW300 (l2)


(g) FT & HW2048 (std. & l2)

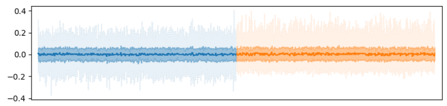
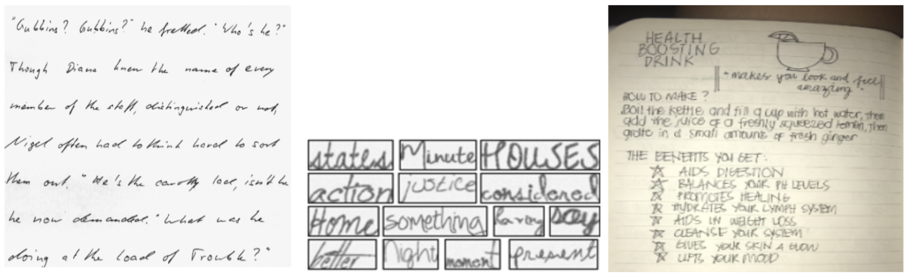

(h) FT & HW300 (std. & l2)

Fig. 3: Mean, standard deviation and dynamic range of the different embedding components for combinations of FastText (FT, blue) and HWNet (orange) with 2048 (HW2048) and 300 (HW300) dimensions. We applied standardization (std.), L2-normalization (l2) and combinations of the both to the embeddings before concatenation. The embeddings were computed on the IAM test set.

instead of only FastText. Another common approach is normalizing the embeddings to have an L2-norm of 1 (see Figs. 3e and 3f). While this normalization has no effect on the cosine similarity of the individual embeddings, changing the length of the vectors before stacking changes the orientation and thus the cosine similarity of the combined embeddings. Additionally, we explore the combination of standardization and length normalization (see Figs. 3g and 3h). In order to examine the influence of the difference in dimensionality of the embedding spaces, we trained the HWNet embeddings to have the same number of dimensions as FastText (300 instead of 2048). The statistics of the combined embeddings using less dimensions are shown in the right column of Fig. 3.

## 4 Experiments

We evaluate our proposed approaches quantitatively as well as qualitatively on three handwriting datasets (see Sec. 4.1). We supply implementation details in section 4.2 and present in section 4.3 the metrics used. In section 4.4, the impact of our proposed normalization steps before concatenating the semantic and syntactic embeddings is evaluated. Finally, we show and discuss the quantitative as well as qualitative results in section 4.5.



(a) IAM-DB [16] (b) HW-Synth12k [12] (c) GNHK [15]

Fig. 4: Example images for the datasets used.

## 4.1 Datasets

In our experiments, we train and evaluate our approaches on three English handwriting datasets (see Fig. 4). The datasets vary considerably in their size and characteristics and include synthetically generated as well as real handwritten documents.

**IAM-DB** The IAM Database [16] is a popular benchmark for handwriting recognition and word spotting. The documents contain modern English sentences and were written by a total of 657 different people. The database includes 1539 text pages containing a total of 13353 text lines and 115320 words. The official writer independent partitioning splits the database in 6161 lines for training, 1840 for validation and 1861 for testing. The pages contain text from a diverse set of categories (e.g. press, religion, fiction).

**HW-Synth12k** The HW-Synth12k (HW) dataset is an in-house version of the HW-Synth dataset proposed in [12]. The dataset consists of synthetically rendered word images from handwritten fonts. The word images contain the 12000 most common words from the English language. For each word, a writer independent split of 50 training and 4 test images is generated. The font is randomly sampled from over 300 publicly available True Type fonts that resemble handwriting.

**GNHK** The GoodNotes Handwriting Kollection (GNHK) dataset [15] includes unconstrained camera-captured images of English handwritten text. It consists of 687 documents containing a total of 9363 text lines and 39026 words. The official partitioning divides the data into training and test sets with a ratio of 75% and 25%, respectively.

### 4.2 Implementation Details

In all of our semantic experiments, we use the pre-trained FastText model proposed by [9]. Similar to Krishnan et al. [14], we normalize the FastText representations to have zero mean and unit variance. For the HWNetv2 model, the PHOC representation consists of layers $2, 3, 4, 5$ and an alphabet with characters $a - z$ and $0 - 9$. The syntactic network follows the proposed optimization and hyper-parameter settings as described in [14].

The semantic network is optimized using the Mean Squared Error Loss and the Stochastic Gradient Descent (SGD) algorithm. The SGD algorithm uses a momentum of 0.9 and a batch size of 64. The networks are first pre-trained on the HW-Synth12k dataset and then fine-tuned on the IAM or GNHK dataset respectively. A learning rate of 0.01 is used during pre-training and 0.001 while fine-tuning. Furthermore, a warm-up strategy is used to train the networks. The images are scaled and padded to a fixed size of $128 \times 384$, while keeping the aspect ratio.

### 4.3 Evaluation Protocol

We use mean Average Precision (mAP) for evaluating the syntactic quality of our approach. MAP is a metric for evaluating retrieval tasks and is de-facto the standard quality measure in the word spotting domain [2,13,21]. In our experiments, we perform evaluation under both segmentation-based QbE and QbS setting. Thereby, we follow the protocol proposed in [2] and discard stop words as queries for the IAM-DB.

Similar to Krishnan et al. [14], we use Word Analogy (WA) for evaluating the semantic quality of our approach. In the WA task, three words $a$, $b$ and $c$ are given and the goal is to infer the fourth word $d$ that satisfies the following condition: $a$ is to $b$ as $c$ is to $d$. For example, *Berlin* is to *Germany* as *Paris* is to *France*. In our evaluation, we use the collection of human-defined WA examples proposed in [19]. Note, that questions which contain words that are not part of the test corpus of a dataset are excluded from the evaluation. The accuracy of correctly predicted analogies is used as the final semantic evaluation score.

### 4.4 Normalization

Figure 5 shows the influence of the weight parameter for the direct combination on the evaluation metrics when applying different normalization methods as explained in Sec. 3.3. The goal of our approach is to allow a user to interpolate between a semantic and syntactic representation as intuitive as possible by adjusting the weight parameter $m$. Therefore, the course of the metrics should be smooth and robust to small changes in the weighting. When combining the embeddings without any normalization (Fig. 5a), we can observe that FastText is only taken into account for very small values of $m$. For larger $m$, the metrics are solely determined by the HWNet embedding. Applying standardization to zero mean and unit variance (Fig. 5b) helps slightly smoothing the metrics

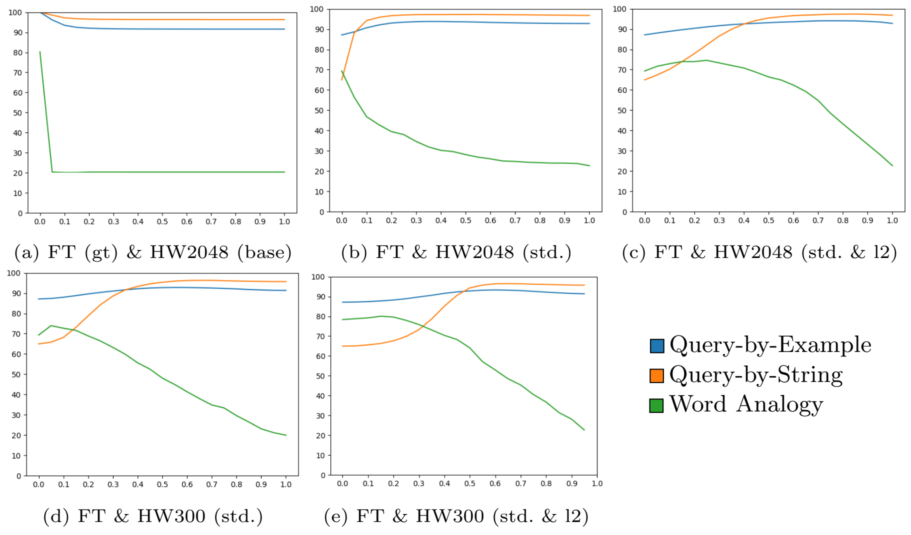
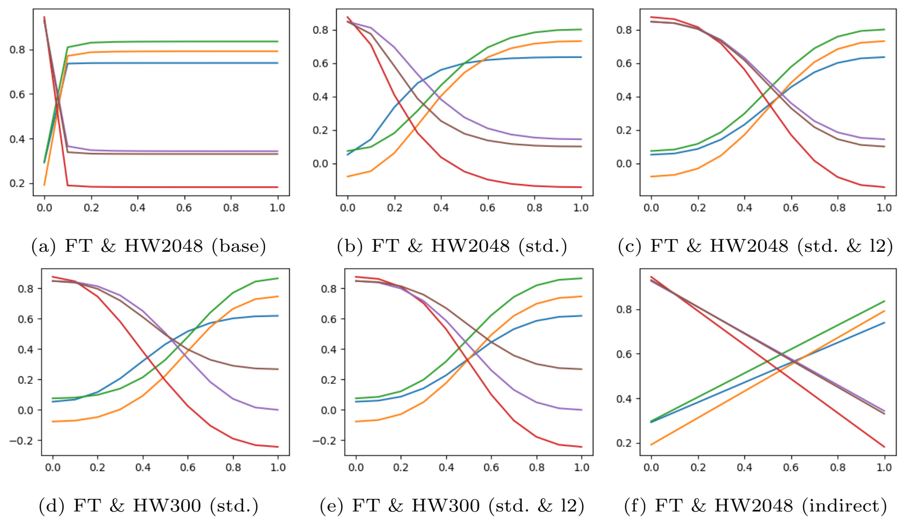


Fig. 5: Impact of the weighted combination for different normalization steps. Results are given in accuracy for WA and mAP for QbE and QbS on the IAM dataset. The values on the x-axis represent the weighting factor $m$ from purely semantic (0) to purely syntactic (1).

as a function of the weight parameter. A further improvement can be achieved by applying L2-normalization after the standardization (Fig. 5c). Reducing the number of dimensions of the HWNet embedding to match the 300 dimensions of FastText allows for a smoother weighting than just combining the original HWNet with 2048 dimensions and FastText (Fig. 5d). However, this combination is still dominated by the HWNet embedding for most choices of $m$. Therefore, we apply standardization and normalization on these embeddings as well (Fig. 5e), achieving a well balanced influence of both embeddings.

A similar effect of the normalization strategies can be observed in Fig. 6 where similarities between example words are depicted depending on the weight parameter. For this example, we chose three syntactically similar (edit distance of 1 or 2) but semantically unrelated words and three semantically similar words which do not share a single character. As a baseline, Fig. 6f shows the indirect approach which is a linear interpolation of the semantic and the syntactic similarity. Figure 6a shows the aforementioned problem of highly unbalanced contributions of the different embeddings towards the similarity between the stacked embeddings when no normalization is applied. Applying standardization (Figs. 6b & 6d) and the combination of standardization followed by L2-normalization (Figs. 6c & 6e) leads to an S-shaped course of the similarity score approximating the desired behaviour of the linear interpolation in the indirect combination. In contrast to the indirect embedding, however, the combined embeddings can be used for other downstream tasks.



(a) FT & HW2048 (base) (b) FT & HW2048 (std.) (c) FT & HW2048 (std. & l2)

(d) FT & HW300 (std.) (e) FT & HW300 (std. & l2) (f) FT & HW2048 (indirect)

🟥 husband & wife 🟪 nine & four 🟫 black & white

🟦 captain & curtain 🟧 sent & seat 🟩 fire & five

Fig. 6: Combined similarities for either syntactically or semantically similar word pairs. The values on the x-axis represent the weighting factor $m$ from purely semantic (0) to purely syntactic (1).

## 4.5 Results

We evaluate our method quantitatively based on the semantic and syntactic metrics described in section 4.3. Since the benefit for a human adaptation of the weighting cannot be easily expressed in a score, we further provide a qualitative evaluation where we show representative retrieval lists for different weightings.

**Quantitative Evaluation** We provide the performances of the direct and indirect weighting approaches in Fig. 7. The plots show the course of the syntactic (QbE and QbS) and semantic (WA) metrics at different weights from purely semantic (0) to purely syntactic (1). Interestingly, the results are very similar for both approaches on all datasets. However, the S-shaped course for the direct method provides a qualitatively improved and more natural ranking. This is also supported by the plots in Fig. 6. In general, the QbS score is strictly monotonically increasing and rises most strongly for weights in the range between 0 and 0.5. After that, only small improvements can be achieved. Furthermore, the QbE score remains fairly constant. The WA score shows a surprising course on the IAM and GNHK datasets, as it rises in the range from 0 to 0.2, drops slightly from 0.2 to 0.5 and finally drops sharply. The increase at the beginning can be explained by the fact that there are a few syntactic analogies in the WA task and they may benefit from the syntactic information.

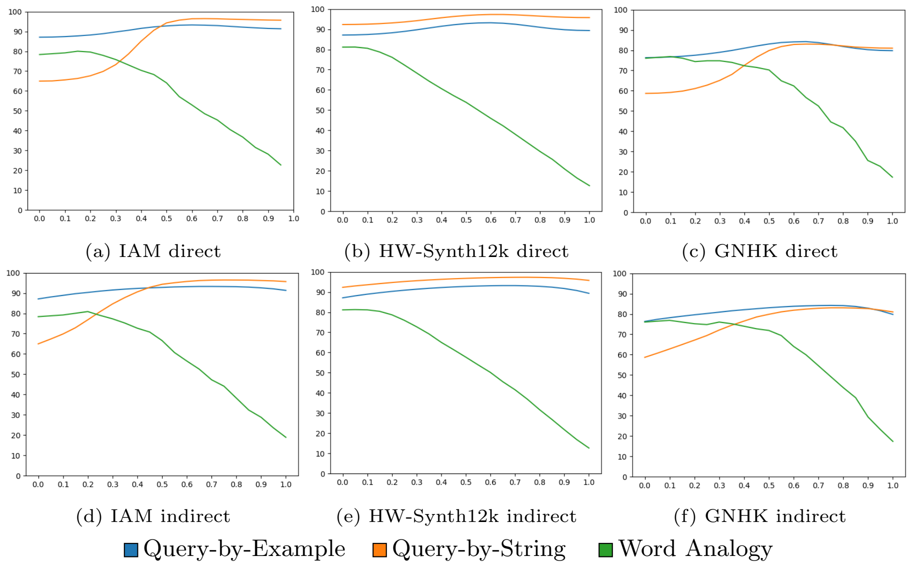


Fig. 7: Impact of the weighted combination for the proposed direct and indirect combination approaches. Results are given in accuracy for WA and mAP for QbE and QbS. The values on the x-axis represent the weighting factor $m$ from purely semantic (0) to purely syntactic (1).

Table 1: Performances on the three evaluated datasets using accuracy for the WA task (semantic) and mAP for QbE and QbS word spotting (syntactic).

| | *HW-Synth12k* | | | *IAM-DB* | | | *GNHK* | | |
|---|---|---|---|---|---|---|---|---|---|
| Approach | WA | QbE | QbS | WA | QbE | QbS | WA | QbE | QbS |
| FastText GT [3] | 82.6 | — | — | 87.4 | — | — | 91.3 | — | — |
| Syntactic (HW300) | 16.4 | 89.4 | 95.8 | 18.9 | 91.3 | **95.7** | 17.4 | 79.9 | **81.0** |
| Semantic (FastText) | **81.2** | 87.3 | 92.3 | **78.4** | 87.1 | 65.0 | **76.0** | 76.3 | 58.7 |
| Krishnan et al. (Stacked) [14] | — | — | — | 61.5 | 90.6 | 94.3 | — | — | — |
| Ours (Indirect) | 57.7 | **92.8** | **96.8** | 66.6 | **92.8** | 94.3 | 72.7 | 82.6 | 78.5 |
| Ours (Direct) | 57.1 | 92.2 | 96.3 | 64.1 | **92.8** | 94.3 | 70.3 | **83.1** | 79.8 |

Table 1 compares our proposed approaches with the recently published stacking method by Krishnan et al. [14]. The main focus of this evaluation is on the comparison of syntactically and semantically combined methods. However, for a better interpretation of the results, we also provide the scores of the purely syntactic and semantic models. Unfortunately, a comparison with [14] is not straightforward, as it is trained and evaluated on an unpublished version of the HW-Synth dataset and no scores have been published for the GNHK dataset.

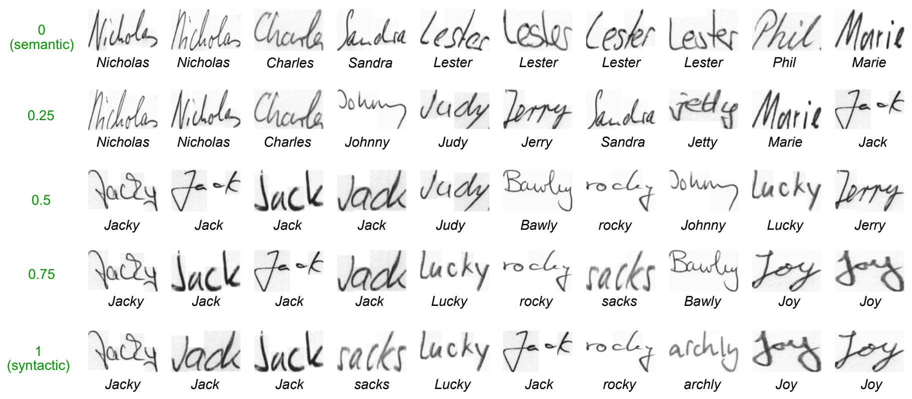

0 (semantic)
Nicholas Nicholas Charles Sandra Lester Lester Lester Lester Phil Marie
0.25
Nicholas Nicholas Charles Johnny Judy Jerry Sandra Jetty Marie Jack
0.5
Jacky Jack Jack Jack Judy Bawly rocky Johnny Lucky Jerry
0.75
Jacky Jack Jack Jack Lucky rocky sacks Bawly Joy Joy
1 (syntactic)
Jacky Jack Jack sacks Lucky Jack rocky archly Joy Joy

Fig. 8: The top-10 of the retrieval lists for the query *jacky* and different weightings of combination on the IAM-DB test set.

Therefore, we can only compare our optimized concatenation approaches on the IAM dataset. The results show that both of our models outperform the approach of Krishnan et al. [14] w.r.t. the WA and QbE metrics and achieve identical scores for QbS. Our methods achieve a proper trade-off between the syntactic and semantic metrics and can enhance the QbE scores remarkably on all datasets. Compared to the textual FastText model, working on the ground truth (GT) annotations of the datasets, the pure semantic model is able to achieve fairly high scores on the IAM and GNHK datasets. However, these are still several points behind the textual approach. For the HW-Synth12k dataset the WA scores are close to the annotation-based system, which is presumably due to the lower variability of the synthesized data and that all words of the test set have been seen during training.

Even though achieving high scores on these metrics is beneficial, the interpretability of the quantitative results is limited. Since the focus of this work is on the realization of an appropriate weighting, the insights from the trajectories given in Fig. 5 and the following qualitative evaluation are increasingly important.

**Qualitative Evaluation** Even though the quantitative results for the combination of syntactic and semantic embeddings could show their ability to perform well on both semantic and syntactic scores, the benefit for the user in exploring document collections cannot be captured that easily. Hence, Fig. 8 and 9 show two examples for the query words *jacky* and *hotel* on the IAM-DB test set with the weightings of (0, 0.25, 0.5, 0.75, 1). The figures illustrate the problem of a purely semantic embedding (0), whereby semantically similar words w.r.t. the query appear in the top 10 of the result lists, but word images with the same transcription as the query are often missing. When syntactic information is taken into account (0.25, 0.5), these words are highly ranked and the list also contains

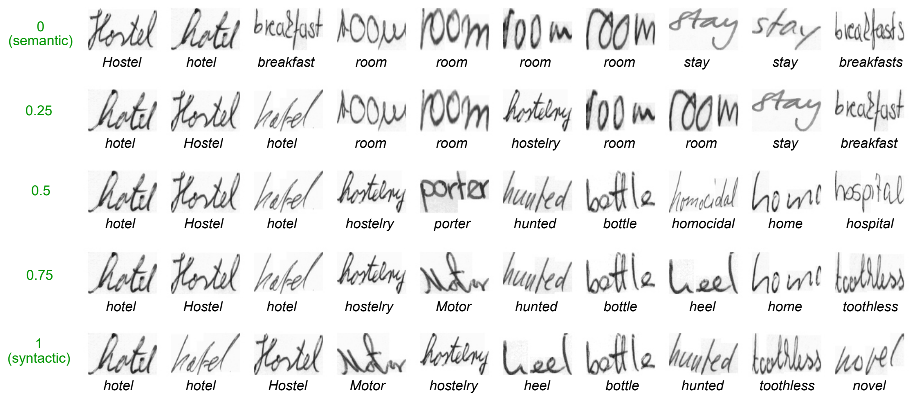

0
(semantic)
Hostel hotel breakfast room room room room stay stay breakfasts
0.25
hotel Hostel hotel room room hostelry room room stay breakfast
0.5
hotel Hostel hotel hostelry porter hunted bottle homocidal home hospital
0.75
hotel Hostel hotel hostelry Motor hunted bottle heel home toothless
1
(syntactic)
hotel hotel Hostel Motor hostelry heel bottle hunted toothless novel

Fig. 9: The top-10 of the retrieval lists for the query *hotel* and different weightings of combination on the IAM-DB test set.

many semantically relevant word images. When the focus shifts more to the syntactic representation (0.75, 1), the words with the same transcription as the query are highly ranked, however, the semantic information is lost. The examples show that a weighted exploration of datasets has many advantages, for example a user searching for the word *hotel* is likely more interested in occurrences of the words *hostel*, *stay* or *room* then to words like *motor* or *heel*.

## 5 Conclusions

In this work, we present and evaluate two approaches for realizing an user-adaptable weighted combination of semantic and syntactic word image representations. Our experiments show both qualitatively and quantitatively that a weighted combination of such representations offers many advantages and yields interesting results especially in the area of semantic word spotting. Although the primary objective of this approach is on the manual adjustment of the weighting parameter by the user, the approach could be extended by an automatic recommendation of this parameter. In future work, we plan to investigate the effect of this combination in the context of recognition-free NLP tasks.

## References

1. Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., Vollgraf, R.: FLAIR: An easy-to-use framework for state-of-the-art NLP. In: NAACL-HLT. pp. 54–59. Minneapolis, MN, USA (2019)
2. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Word spotting and recognition with embedded attributes. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(12), 2552–2566 (2014)
3. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the ACL **5**, 135–146 (2017)

4. Dang, H., Le-Hong, P.: A combined syntactic-semantic embedding model based on lexicalized tree-adjoining grammar. Comput. Speech Lang. **68**, 101202 (2021)
5. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: ACL. pp. 4171–4186. Minneapolis, MN, USA (2019)
6. Ethayarajh, K.: How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In: Proc. Conf. on Empirical Methods in Natural Language Processing. pp. 55–65. Hong Kong (2019)
7. Giotis, A.P., Sfikas, G., Gatos, B., Nikou, C.: A survey of document image word spotting techniques. Pattern Recognition **68**, 310–332 (2017)
8. Gordo, A., Almazán, J., Murray, N.: LEWIS: Latent embeddings for word images and their semantics. In: ICCV. Santiago, Chile (2015)
9. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning word vectors for 157 languages. In: Proc. Int. Conf. on Language Resources and Evaluation. Miyazaki, Japan (2018)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778. Las Vegas, NV, USA (2016)
11. Krishnan, P., Jawahar, C.V.: Bringing semantics in word image retrieval. In: ICDAR. pp. 733–737. Washington, DC, USA (2013)
12. Krishnan, P., Jawahar, C.V.: Generating synthetic data for text recognition. CoRR **abs/1608.04224** (2016)
13. Krishnan, P., Jawahar, C.V.: HWNet v2: An efficient word image representation for handwritten documents. IJDAR **22**, 387–405 (2019)
14. Krishnan, P., Jawahar, C.: Bringing semantics into word image representation. Pattern Recognition **108**, 107542 (2020)
15. Lee, A.W.C., Chung, J., Lee, M.: GNHK: A dataset for English handwriting in the wild. In: ICDAR. pp. 399–412. Lausanne, Switzerland (2021)
16. Marti, U., Bunke, H.: The IAM-database: An English sentence database for offline handwriting recognition. IJDAR **5**(1), 39–46 (2002)
17. Mathew, M., Gómez, L., Karatzas, D., Jawahar, C.V.: Asking questions on handwritten document collections. IJDAR **24**, 235–249 (2021)
18. Memon, J., Sami, M., Khan, R.A., Uddin, M.: Handwritten optical character recognition (OCR): A comprehensive systematic literature review. IEEE Access **8**, 142642–142668 (2020)
19. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR. Scottsdale, AZ, USA (2013)
20. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: NAACL. pp. 2227–2237. New Orleans, LA, USA (2018)
21. Sudholt, S., Fink, G.A.: PHOCNet: A deep convolutional neural network for word spotting in handwritten documents. In: ICFHR. pp. 277—282. Shenzhen, China (2016)
22. Tüselmann, O., Wolf, F., Fink, G.A.: Identifying and tackling key challenges in semantic word spotting. In: ICFHR. pp. 55–60. Dortmund, Germany (2020)
23. Wang, X., Jiang, Y., Bach, N., Wang, T., Huang, Z., Huang, F., Tu, K.: Automated concatenation of embeddings for structured prediction. In: ACL/IJCNLP. pp. 2643–2660. Bangkok, Thailand (2021)
24. Wilkinson, T., Brun, A.: Semantic and verbatim word spotting using deep neural networks. In: ICFHR. pp. 307–312. Shenzhen, China (2016)