Identifying and Tackling Key Challenges in Semantic Word Spotting

Oliver Tüselmann
Department of Computer Science
TU Dortmund University
44221 Dortmund, Germany
oliver.tueselmann@cs.tu-dortmund.de

Fabian Wolf
Department of Computer Science
TU Dortmund University
44221 Dortmund, Germany
fabian.wolf@cs.tu-dortmund.de

Gernot A. Fink
Department of Computer Science
TU Dortmund University
44221 Dortmund, Germany
gernot.fink@cs.tu-dortmund.de

***Abstract*—Semantic word spotting is an extension of the traditional word spotting approach that uses not only visual but also semantic information to determine the similarity between a word image and a given query. Current approaches in this area achieve a semantic retrieval by embedding word images into a textually trained semantic space. The related literature presents remarkable results regarding established metrics indicating that the task of semantic word image retrieval is solved. A closer look at the results reveals, however, that this is only partially the case. In this work, we identify and solve current key challenges for semantic word spotting. We analyze the published works in this field towards these challenges and show why they do not solve them. For this purpose, we demonstrate that the used embedding space from current methods contains strong artifacts influencing the retrieval task. Furthermore, we evaluate a more suitable and established embedding approach from Natural Language Processing for semantic word spotting. We also explain the challenges of mapping word images into a semantic embedding space and evaluate different architectures for this task. Thereby, we present a new architecture that outperforms current approaches in this area. In addition, we show that commonly used metrics are not suitable for evaluating a semantic retrieval and present a new evaluation metric for this task.**

I. Introduction

In recent years, there has been an increased interest in tasks related to text recognition and retrieval for handwritten documents. Due to the high variability in handwriting, it is still an open research topic. As the recognition of these challenging documents is not always easy to obtain, the interest in more robust approaches such as word spotting increases [1]. The goal of this task is to retrieve the most relevant instances of words in a collection of scanned documents w.r.t. a query. 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 [1]. Recently, approaches based on Convolutional Neural Networks (CNNs) have achieved remarkable results for most benchmark datasets in this area [2], [3]. For state-of-the-art word spotting methods, only instances with the same transcription as the query are considered relevant. Therefore, a search for the word ”dog” would not show any result for ”dalmatian”, despite being relevant to a user. A procedure that has already significantly improved the user satisfaction for web search engines is the consideration of semantic information in the retrieval. This allows users to not only search for 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*).

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. Due to a different interpretation of semantic similarity between words, the publications in this field can be divided into *ontology-based* and *context-based* semantic retrieval. In ontology-based approaches, the semantic relationship between words is based on categories [4], [5]. These relationships are mostly organized in a handcoded graph like WordNet [6]. In context-based approaches the semantic relationships between words rely on the distributional hypothesis. This states that words appearing in the same context have a similar meaning. There are many models from Natural Language Processing (NLP) that can extract these similarities between words [7]–[9]. State-of-the-art approaches for context-based semantic word spotting obtain the semantic information of a word image by embedding the image into a textually trained semantic space [2]. Based on the published retrieval lists and the remarkable results regarding common metrics, it seems that the approach from Wilkinson et al. [2] solves the task of semantic word image retrieval. A closer look at the results, however, shows that their approach can only partially solve it.

In this work, we identify the current key challenges for context-based semantic word spotting on segmented word images. We also analyze the published works in this field towards these challenges and show why they are not able to solve them. Furthermore, we present solutions for two of

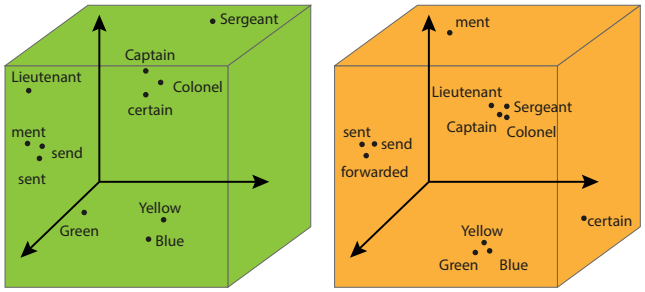

Fig. 1: Comparison between the learned semantic spaces defined by the CharLSTM (green) and FastText (orange) model.

these challenges. The remainder of this paper is structured as follows. In section II, we compare the word embedding model used in [2] to a well known approach from NLP. In section III, we explain the challenges of mapping word images into a semantic embedding space and evaluate different architectures. In section IV, we show that commonly used metrics are not suitable for the evaluation of a semantic retrieval list and present a new evaluation metric for semantic word spotting.

II. Word Embeddings

Word embeddings are a well known approach in NLP to convert strings into a vector representation. They are already used in word spotting systems (e.g. *Pyramidal Histogram of Characters (PHOC)* and *Discrete Cosine Transform of Words*) to enable a comparison between word images and strings. Thereby, the choice of the embedding approach has a strong influence on retrieval [2], [3]. This is particularly true for context-based semantic word spotting, since the embedding approach is also used to encode semantic relationships between words. Therefore, it is a major challenge to find or create a suitable word embedding model for this task.

Due to the recent progress in NLP, there are several promising models to choose from. However, not every model is directly applicable to the problem. This is especially the case for state-of-the-art approaches, as they require context information that is not available for segmentation-based word spotting. Furthermore, the approach should not generate an embedding for each word independently (e.g. Word2Vec [10]), but provide a learnable structure for encoding words. This may allow a word spotting system to extract a structure from embeddings already seen during training. Based on this structure, the correct embedding can then be predicted for words that were not part of the training. This is very important because otherwise the word spotting system can only determine the correct embedding for words that were part of the training set.

The semantic word spotting system from Wilkinson et al. [2] uses the Character-Aware Neural Language Model (CharLSTM) [9]. This model seems to work fine for this task, because it offers a learnable structure and does not need context information to produce embeddings. In this model, several steps are performed to calculate a vector representation

TABLE I: Most similar words from GW and IAM dataset w.r.t. *captain* and *sent* using the CharLSTM model. Values in brackets show the cosine similarity w.r.t the query times 100.

GW		IAM	
captain (100)	***sent*** (100)	***captain*** (100)	***sent*** (100)
captains (75)	send (47)	curtain (62)	cent (63)
cap (39)	ment (38)	certain (57)	ment (62)
tain (34)	sen (37)	captain's (56)	went (55)
colonel (34)	sending (37)	explain (44)	set (55)
subaltern (33)	set (30)	entertain (42)	event (54)
adjutant (33)	getting (25)	acceptable (42)	send (53)
lieutenant (32)	tents (25)	ception (41)	spent (50)
major (32)	gene (24)	uncertain (41)	sect (45)
certain (30)	returned (24)	casting (39)	seat (44)

TABLE II: Most similar words from GW and IAM dataset w.r.t. *captain* and *sent* using the FastText model. Values in brackets show the cosine similarity w.r.t the query times 100.

GW		IAM	
captain (100)	***sent*** (100)	***captain*** (100)	***sent*** (100)
captains (78)	send (74)	commander (62)	send (74)
sergeant (53)	sending (71)	sailor (57)	received (63)
colonel (51)	forwarded (68)	admiral (56)	returned (57)
officer (51)	received (63)	coach (51)	arrived (51)
ensign (51)	returned (57)	officer (51)	came (51)
lieutenant (49)	delivered (55)	crew (50)	brought (51)
chief (46)	arrived (51)	ship (49)	mail (48)
soldier (44)	came (51)	lieutenant (49)	receive (48)
adjutant (42)	ordered (51)	chief (46)	letter (47)

for a given word. The first step is to transform the word into a character embedding matrix. Then multiple filters of different sizes extract n-gram-like information from it. By applying a *Max-Over-Time Pooling* operation to these features, a fixed length representation is created. The resulting vector can be used as an independent representation or can be transformed into a semantic space by using a highway network [11]. In the training phase, a distribution over the vocabulary is predicted by using the determined word representation and a *Recurrent Neural Network Language Model*. Finally, the *Cross Entropy Loss* between the predicted distribution and the next word from the training is minimized. As there are no pre-trained models available, the authors of [2] suggested, that the model could be trained on all the 40 Volumes of *"The Writings of George Washington from the original Manuscript Sources 1745-1799"* for the George Washington (GW) dataset [12]. For the IAM database [13], they suggest to use the text corpus of the Penn Treebank. To obtain an impression of the semantic spaces generated by the CharLSTM model, table I lists the ten most similar words for two exemplary queries w.r.t. the model. The analysis of the spaces shows that there are qualitatively good semantic relations for some words, especially for the GW dataset (e.g. *captain*). But it is obvious that for most queries there are artifacts in the spaces that lead to a high ranking for orthographically but not semantically similar words. In case of the IAM database, the artifacts become even more obvious and apply to almost all query words.

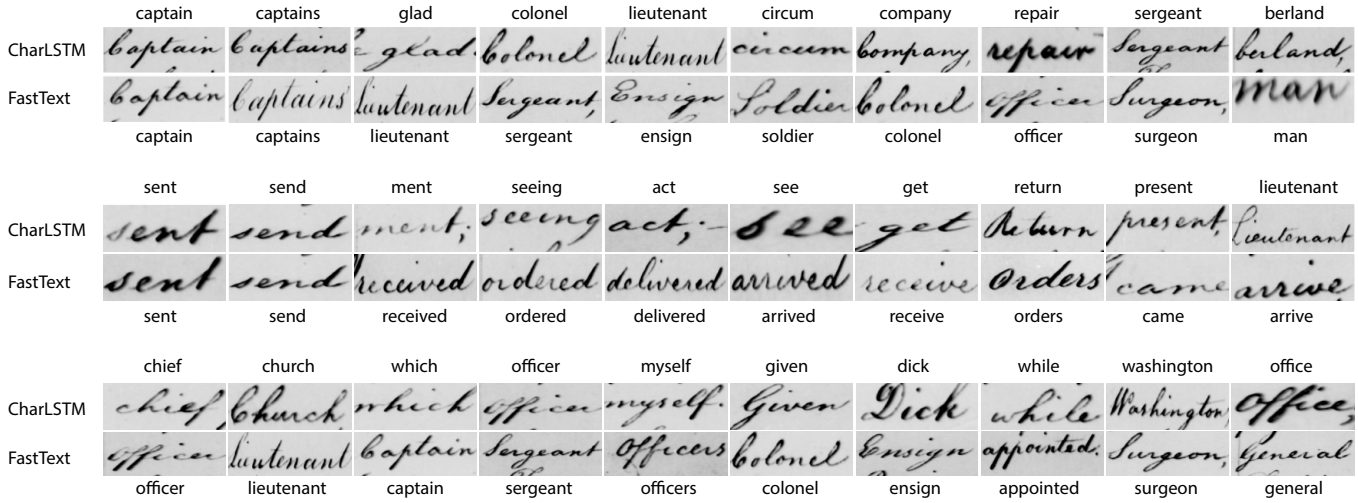

Fig. 2: Comparison between CharLSTM and FastText. The figure depicts the top ten unique results for the queries *captain* (top), *sent* (middle) and *chief* (bottom). The retrieval lists were generated using the approach of [2] and the GW test dataset.

To analyze the influence of the semantic embedding on the retrieval, we also use the FastText [8] model. This approach seems to be well suited for this task, because it has a learnable structure, established pre-trained models and does not need context information to produce embeddings. Figure 1 visualizes both embedding spaces for exemplary words. The comparison indicates that FastText provides a qualitatively better semantic embedding that is more independent from the word's orthography. The FastText approach uses a Neural Network and an unstructured training corpus to learn a mapping of words to real vectors. Instead of learning an embedding for each word independently, it uses the internal structures of words. This is achieved by representing each word as a composition of n-grams. There are also special boundary symbols $<$ and $>$ at the beginning and end of words, allowing to distinguish prefixes and suffixes from other character sequences. The word itself is also included in the set of its n-grams. Taking the word *where* and $n = 3$ as an example, it will be represented by the character n-grams: $<$*wh, whe, her, ere, re*$>$ and the special sequence $<$*where*$>$. For training, either the Skip-gram model or the Continous Bag-of-Words model can be used. The overall goal of training is to produce similar embeddings for words that occurred in the same context. A main reason for using FastText is the availability of publicly released pre-trained models [14]. We use the model trained on *Common Crawl* and *Wikipedia* for the English language. Table II offers a first impression regarding the high semantic quality of this model. The effect of using an embedding space with qualitatively better semantic relationships for word spotting can be seen in figure 2. It shows a qualitative comparison of the generated semantic retrieval lists on GW using the CharLSTM and FastText model.

III. Word image mapping

Semantic word spotting exploits the encoded knowledge available in the textual domain for retrieval. Since semantic information is directly predicted from a word image without transcribing it, a mapping of word images to the representation of their transcriptions in the semantic space is required. State-of-the-art approaches in traditional word spotting achieve remarkable results by using models based on CNNs to learn an embedding for word images [3]. This is mainly due to the strong relationship between the distances of word images in the embedding space and the edit distance of their transcriptions. In the case of a semantic embedding, however, this property is not given and word images with a small edit distance w.r.t. their transcriptions often have to be mapped into very different areas of the embedding space. The challenge of mapping word images to a semantic space can be seen in figure 3. Here, it is shown that similar-looking word images are close to each other in an embedding space traditionally used for word spotting, whereas in a semantic space they are often far away from each other. Although almost all word images have the transcription *fire*, most of them are mapped to different words in the semantic space due to their visual appearance. Therefore, they are located far away from the target representation in the semantic space. As a result, these word images have a lower ranking for the query word *fire*. Whereas in traditional word spotting, all the word images from the example are still quite close to the target representation, even if they are mapped incorrectly. This is why they are still having a high ranking in the retrieval list of *fire*.

Instead of learning an embedding for each word directly, suitable NLP models for semantic word spotting divide words into their n-gram representations in order to determine their embeddings. This approach allows the mapping function to calculate an embedding even for unseen words during training. Even though this is a very useful property, it is still a challenge for the model to not only learn an embedding for word images used in training, but to generalize and approximate the learnable embedding function used in the NLP model.

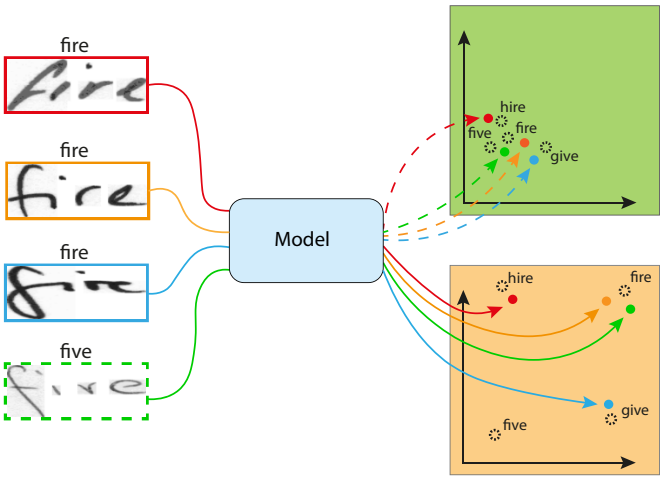

Fig. 3: Visualization of the challenges for mapping word images into semantic space (orange) and compares them to traditional embedding space (green). The dashed dots indicate the correct position of the corresponding word in the respective space and the dashed rectangle shows the word images whose transcription did not occur during training.

If the model cannot approximate the underlying embedding function accurately enough, it can only predict the correct representation for transcriptions occurring in training. It can be seen that this kind of function very often maps word images with unseen transcriptions in completely wrong areas of the semantic space. This problem is visualized in figure 3, where the green highlighted word image with the transcription *five* was not a part of training and is therefore incorrectly mapped to the representation of the orthographically similar word *fire*.

The semantic word spotting approach in [2] uses a two-stage architecture to realize a mapping from word images to a semantic space. The first step is to transform word images into image descriptors by using a TripletCNN. This approach tries to ensure that word images with the same transcription are mapped to similar, and word images with different transcriptions to dissimilar vector representations. The second and final step is to map these descriptors into a semantic space by using a fully connected Multilayer Perceptron (MLP). We reproduced their approach with the suggested training procedure and hyperparameters. Thereby, we achieved higher mAP scores for the CharLSTM embeddings. This can be explained by the different pre-processing of the training data for the CharLSTM models. In the following, we will refer to the reproduced model as TripletCNN + MLP. In table III it can be seen that the architecture produces satisfying results for the CharLSTM embedding. This may be explained by the high similarity between the CharLSTM and an orthographic space, as this avoids the problems of mapping a word image into a fully semantic space. By using this architecture for a semantically better space (e.g. FastText), it becomes clear that it cannot solve the problems described above. This cannot only be illustrated by the example of figure 2 for the query word *chief*, but also by the results in table III. A feasible explanation for this is that the extracted image descriptors from a TripletCNN strongly restricts the ability of mapping word images into a semantic space. Even if the descriptors calculated by the TripletCNN are effective in distinguishing word images with different transcriptions, they do not take orthographic information into account. Since the NLP methods used to determine word embeddings are based on n-gram information, it becomes difficult or even impossible for the MLP to extract the correct mapping function.

In order to avoid the possible drawbacks of the TripletCNN, this work evaluates two architectures for semantic word spotting. We first evaluate the end-to-end architecture proposed by Sudholt et al. [3] that yields state-of-the-art results for traditional word spotting. Although the architecture seems to be well suited for semantic word spotting, it could not solve the identified problems and even performs worse (see table III). Since the end-to-end principle has not been able to achieve satisfying results, the use of a two-stage approach seems to be preferable for this task. To better approximate the embedding function of the NLP models, we replace the image descriptors obtained from the TripletCNN by a representation that approximates the orthographic information. An embedding that fulfills these conditions is the PHOC representation. Therefore, we replace the TripletCNN in the architecture of [2] by the TPP-PHOCNet from Sudholt et al. [3]. For training, we use the standard configurations published in the respective works. The only differences are that we pre-train the PHOCNet with the IIITHWS dataset [15] and freeze the weights of the PHOCNet when training the MLP.

Table III shows, that our new architecture outperforms state-of-the-art results for semantic word spotting. By analyzing the QbS values for the CharLSTM embedding, it becomes obvious that the model from [2] cannot approximate the mapping functions properly. Whereas our architecture can learn the structure almost perfectly and therefore there is only a small difference between the QbS values of all queries and those that only occurred during training. The major difference between these values in the FastText model is that the additional n-gram is dominant for the construction of the representation. Thus, the encoding of the additional n-gram is needed in order to reconstruct the representation. The FastText structure is not as learnable as expected because it is not possible to compute the representation of a word from the vocabulary by just summing up his n-gram representations. Therefore, words that do not occur in the MLP training are mapped far away from the target representation. Furthermore, it can be seen that the CharLSTM embedding yields higher QbS and QbE values than FastText.

IV. Evaluation

Mean Average Precision (mAP) has traditionally been used as a measure for the evaluation of retrieval lists in word spotting. This metric is well suited for a traditional approach, since it is only important that the word images with the same transcription w.r.t. the query are highly ranked. Secondary retrievals (i.e. all word images with a different transcription

TABLE III: Comparison of the CharLSTM and FastText model on the GW dataset and IAM database for the three architectures. The values in brackets show the mSP using only those words and word images that were part of the training as queries. Results reported as mean Average Precision [%].

		GW		**IAM**	
Embedding	Architecture	QbE	QbS	QbE	QbS
CharLSTM	TripletCNN + MLP	**97.55** (**97.82**)	83.89 (98.10)	82.57 (85.84)	80.96 (91.32)
	AttributeCNN	92.31 (94.01)	76.46 (94.54)	65.57 (63.71)	75.91 (79.14)
	PHOCNet + MLP	94.94 (96.82)	**94.33** (**98.66**)	**85.93** (**86.94**)	**88.62** (**91.45**)
FastText	TripletCNN + MLP	94.12 (95.07)	57.42 (85.54)	51.01 (63.01)	19.29 (44.52)
	AttributeCNN	87.58 (88.13)	59.39 (85.58)	10.02 (12.07)	12.26 (20.90)
	PHOCNet + MLP	96.45 (96.56)	69.14 (98.32)	81.61 (86.58)	54.41 (91.66)

than the query), however, are completely ignored. Since the focus of semantic word spotting is also on the ranking of the secondary retrievals, the use of this metric is not suitable for the evaluation of this task. This can be illustrated by using figure 2 as an example, where all retrieval lists except the last one have a mAP of 1.0 but a totally different semantic quality. The misuse of the mAP score for the evaluation of semantic retrieval lists is also evident in the evaluation of the semantic models in section III. Here the CharLSTM has a high mAP score for both the GW dataset and the IAM database, although the secondary retrievals are mostly semantically dissimilar. For FastText the mAP scores are lower, although the secondary retrievals show a high semantic similarity. For this reason we present and evaluate a more suitable metric for semantic word spotting in this section.

For evaluating a semantic retrieval list, the semantic similarities between the query and the elements of the retrieval list are required. A fundamental question here is how semantic relations between words can be quantified. One possibility that is explored in this work is to use a semantic word model, such as FastText or the CharLSTM, to determine the semantic similarities between the words. The proposed evaluation measure extends the idea of the Average Precision (AP) by considering semantic information in the evaluation of the retrieval order. The AP is a measure for evaluating the sorting of a retrieval list and can be formally calculated, for a list with the length n, by

$$AP = \frac{\sum_{k=1}^{n} p(k) \cdot r(k)}{t}. \tag{1}$$

Here $r(\cdot) \in \{0, 1\}$ is an indicator function with $r(i)$ being 1 if the transcription of the i-th element in the retrieval list is equal to the query and 0 otherwise. Furthermore $t = \sum_{i=1}^{n} r(i)$ is the number of word images in the list with the same transcription w.r.t. the query. Equation 2 determines the precision for the retrieval list up to position k. This score describes the ratio between the number of relevant and irrelevant items in the list. The number of relevant items is calculated by summing up the values $r(1)$ to $r(k)$.

$$p(k) = \frac{\sum_{i=1}^{k} r(i)}{k} \tag{2}$$

In the following, the AP formula is adapted to the evaluation of semantic retrieval lists. For this purpose, it is obvious that, unlike AP, there is no 1 or 0 decision on whether a word is semantically similar to the query or not. It is rather the case that every word in the list has a semantic relationship regarding the query and therefore all elements of the list are relevant for the evaluation. Therefore, the indicator function $r(i)$ is replaced by the function $sim(a, i)$. This function uses a semantic model to determine semantic similarity between a query a and the word at position i in the retrieval list. This modification in the precision formula results in the so-called *similarity* or short *s* score, which is formally calculated by

$$s(a, k) = \frac{\sum_{i=1}^{k} sim(a, i)}{k}. \tag{3}$$

Instead of the precision $p(k)$ the $s(a, k)$ score determines the ordering of the semantic similarities up to the position k in the list. The replacement of these functions in the numerator of the AP formula yields to the so-called *Semantic Precision* or short *SP*, which can be formally determined for a list l of size n by

$$SP(a, l) = \sum_{k=1}^{n} s(a, k) \cdot sim(a, k). \tag{4}$$

For the interpretation of this evaluation metric, it is important to normalize the scores to the interval $[0, 1]$. However, normalizing the SP for a list is not as straightforward as using $t = \sum_{k=1}^{n} sim(a, k)$. In order to achieve this, the semantically best possible ranking for a query is needed, which can be obtained by sorting the word images contained in the database in descending order of their semantic similarity to the query. In a traditional segmentation based word spotting approach all elements of the database are part of the retrieval list, which makes it possible to calculate the best retrieval list by just sorting it w.r.t. the similarity scores. Finally, the normalization of the SP for the retrieval list can be achieved by dividing it with the SP of the best possible ranking for the query. This results in the complete definition of the normalized *Semantic Precision*, which can be formally calculated for the retrieval list $l_{retrieval}$, a query a and the best possible retrieval l_{best} by

$$SP_{norm}(a, l_{retrieval}) = \frac{SP(a, l_{retrieval})}{SP(a, l_{best})}. \tag{5}$$

Averaging the normalized SP scores for all queries from the test set leads to the definition of the *mean Semantic Precision (mSP)*.

TABLE IV: Evaluation results for the two semantic models and PHOC as baseline with mSP. The values in brackets show the mSP using only those words and word images that were part of the training as queries. Results reported as mSP [%].

		GW		IAM	
Dataset	Architecture	**mSP_{all}**	**mSP_{10}**	**mSP_{all}**	**mSP_{10}**
PHOC	TripletCNN + MLP	50.58 (56.66)	53.61 (59.67)	42.83 (56.81)	45.67 (58.93)
	AttributeCNN	51.59 (57.61)	55.52 (60.08)	48.34 (58.35)	50.48 (59.71)
	PHOCNet + MLP	51.39 (57.46)	54.88 (59.87)	48.02 (57.89)	50.31 (58.49)
CharLSTM	TripletCNN + MLP	53.89 (61.86)	57.79 (68.48)	45.91 (63.08)	48.44 (65.95)
	AttributeCNN	52.66 (53.61)	56.66 (59.67)	49.08 (66.84)	45.32 (59.94)
	PHOCNet + MLP	54.79 (62.18)	60.27 (70.53)	49.19 (61.06)	52.55 (63.00)
FastText	TripletCNN + MLP	76.87 (85.81)	71.06 (87.52)	54.82 (73.76)	39.04 (62.13)
	AttributeCNN	60.12 (70.44)	59.72 (75.15)	54.22 (69.72)	26.68 (33.12)
	PHOCNet + MLP	**78.46** (**89.52**)	**75.50** (**94.25**)	**73.29** (**87.74**)	**69.82** (**90.90**)

To demonstrate the applicability of the new metric for semantic word spotting, we present the mSPs for the two semantic models on the GW dataset and the IAM database. We use a pre-trained FastText model to obtain the semantic similarities for the metric. Thereby, it is important to note that we do not use the same pre-trained FastText model to avoid any bias. The results are shown in table IV where not only the mSP is calculated for the whole retrieval list (mSP_{all}) but also for the first ten (mSP_{10}) elements. We further report values for using all words as queries and for only using those that were part of training. First of all, it is obvious that the quality of the retrieval lists for queries that were part of the training is significantly better than for queries that did not occur in training. The PHOC representation can be considered a baseline as it does not encode semantic information. Contrary, it is obvious that the CharLSTM, unlike the PHOC, encodes semantic information which is also reflected in the mSP results, as the CharLSTM has higher scores for this metric. It is also evident that the FastText embedding achieves by far the best results. Furthermore, this evaluation supports the observations from our analysis and shows that the FastText model encodes significantly more semantic information than the CharLSTM. Another indicator for the high semantic quality of the FastText model is that even after the first ten results there are highly ranked semantic word images. This can be seen by comparing the mSP_{all} and mSP_{10} values. For the PHOC and CharLSTM the mSP_{all} values decreases.

V. Conclusion

In this work, we highlight the advantages of using semantic information in word spotting approaches and identify the current key challenges for this task. Our analysis shows that the choice of the semantic embedding model has a major influence on retrieval. Therefore, the first key challenge is to find or create a semantic word embedding that offers a learnable structure, encodes relevant semantic information and does not need context information. We show that the CharLSTM embedding used in [2] has a strong relationship to a orthographic space and does not encode many semantic relationships. We also show that the FastText approach offers a better semantic space but is not as learnable as expected. Therefore, further research in this area is needed to overcome this challenge. The second key challenge is the mapping of word images into a semantic space. We show that the approach from [2] has a major drawback for semantic word spotting and evaluate an end-to-end approach which unexpectedly performs worse. However, we are able to tackle the challenge by replacing the TripletCNN in the architecture of [2] with a TPP-PHOCNet. We also demonstrate that the current metrics in word spotting are not suitable for evaluating semantic retrieval. Therefore, the third key challenge is to find or create a metric for semantic word spotting. In this work, we introduce the *mean Semantic Precision* for this purpose and show that this metric is able to reasonably score a semantic ranking.

References

[1] A. P. Giotis, G. Sfikas, B. Gatos, and C. Nikou, "A survey of document image word spotting techniques," in *Pattern Recognition*, vol. 68, 2017, pp. 310–332.

[2] T. Wilkinson and A. Brun, "Semantic and verbatim word spotting using deep neural networks," in *ICFHR*, Oct 2016, pp. 307–312.

[3] S. Sudholt and G. A. Fink, "Attribute CNNs for word spotting in handwritten documents," in *IJDAR*, vol. 21, no. 3, 2018, pp. 199–218.

[4] A. Gordo, J. Almazan, and N. Murray, "LEWIS: Latent embeddings for word images and their semantics," in *ICCV*, 2015.

[5] P. Krishnan and C. V. Jawahar, "Bringing semantics in word image retrieval," in *ICDAR*, Aug 2013, pp. 733–737.

[6] C. Fellbaum, *WordNet: An Electronic Lexical Database*, ser. Language, Speech, and Communication. Cambridge, MA: MIT Press, 1998.

[7] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL*, 2018.

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," in *Trans. of the Association for Computational Linguistics*, vol. 5, 2017, pp. 135–146.

[9] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," in *Proc. AAAI Conf. on Artificial Intelligence*, 2016, pp. 2741–2749.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Int. Conf. on Neural Information Processing*, vol. 2, 2013, pp. 3111–3119.

[11] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2377–2385.

[12] T. M. Rath and R. Manmatha, "Word spotting for historical documents," in *IJDAR*, vol. 9, no. 2-4, 2007, pp. 139–152.

[13] U.-V. Marti and H. Bunke, "The IAM-database: an English sentence database for offline handwriting recognition," in *IJDAR*, vol. 5, no. 1, Nov 2002, pp. 39–46.

[14] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," in *Proc. of the Int. Conf. on Language Resources and Evaluation*, 2018.

[15] P. Krishnan and C. V. Jawahar, "Hwnet v2: an efficient word image representation for handwritten documents," *IJDAR*, vol. 22, pp. 387–405, 2019.