Papy-S-Net : A Siamese Network to match papyrus fragments

Antoine Pirrone, Marie Beurton-Aimar, Nicholas Journet

Papy-S-Net : A Siamese Network to match papyrus fragments

Antoine Pirrone*
antoine.pirrone@labri.fr
Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800
Talence, France

Marie Beurton Aimar
marie.beurton@labri.fr
Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800
Talence, France

Nicholas Journet
journet@labri.fr
Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800
Talence, France

ABSTRACT

Like all heritage documents, papyri are the subject of an in-depth study by scientists. While large volumes of papyri have been digitized and indexed, many are still waiting to be so. It takes time to study a papyrus mainly because they are rarely available in one piece. Papyrologists must review a large number of fragments, find those that go together and then assemble them to finally analyze the text. Unfortunately, some fragments no longer exist. It is then a time consuming puzzle to solve, where not all the pieces are available and where fragments boundaries are not perfectly matching.

This article describes a method to help Papyrologists save time by helping them to solve this complex puzzle. We provide a solution where an expert use a fragment as a request element and get fragments that belong to the same papyrus. The main contribution is the proposal of a deep siamese network architecture, called **Papy-S-Net** for Papyrus-Siamese-Network, designed for papyri fragment matching. This network is trained and validated on 500 papyrus fragments approx. We compare the results of **Papy-S-Net** with a previous work of Koch et al. [14] which proposes a siamese network to match written symbols. In order to train and validate the network, we proceed to the extraction of patches from the papyrus fragments to create our ground truth. **Papy-S-Net** outperforms Koch et al.'s network. We also evaluate our approach on a real use case on which **Papy-S-Net** achieves 79% of correct matches.

CCS CONCEPTS

• **Applied computing** → **Document Analysis**; • **Computing methodologies** → *Neural networks.*

KEYWORDS

Historical documents, Deep Learning, Siamese network

ACM Reference Format:
Antoine Pirrone, Marie Beurton Aimar, and Nicholas Journet. 2019. Papy-S-Net : A Siamese Network to match papyrus fragments. In *The 5th International Workshop on Historical Document Imaging and Processing (HIP '19), September 20–21, 2019, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3352631.3352646

*All authors contributed equally to this research.

HIP '19, September 20–21, 2019, Sydney, NSW, Australia

ACM ISBN 978-1-4503-7668-6/19/09…$15.00
https://doi.org/10.1145/3352631.3352646

1 INTRODUCTION

The papyri studied here are coming from Egyptian mummies. These documents were written around the third century BCE and their supports were afterwards reused to make mummy casings called "Cartonnages". These casings were painted funeral ornaments decorating the mummies. A funeral mask made up of several dozen papyri fragments can be seen on the left of Figure 3. Papyrus being an expensive and rare resource at that time, these ornaments were made with documents that did not bear interest anymore since they were already covered with writing.

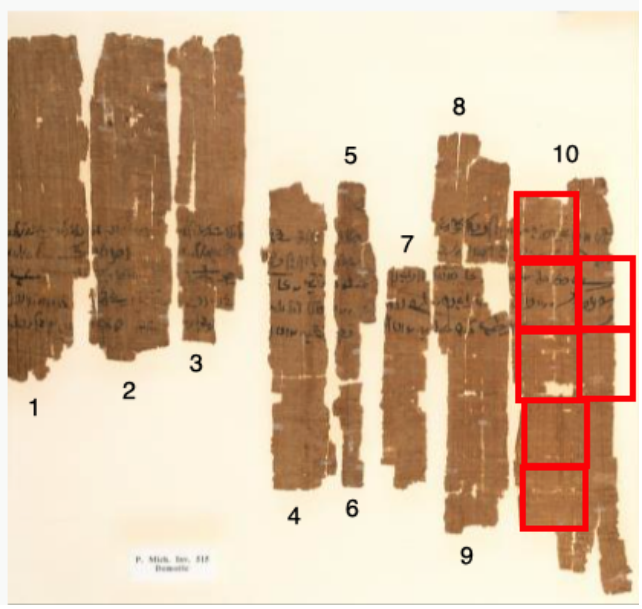


Figure 1: Definition of terms related to the study of papyri. This image shows a papyrus composed of 10 fragments. On each fragment we extract patches that will be used to train our network. For example on fragment 10 we extract 7 patches

Modern Papyrologists are interested in studying the content of these papyri in order to obtain information on the economic activity of the region from which they come : agricultural production, tax policy …In order to analyse their textual content, the ornaments were in the past torn apart but are nowadays carefully separated. Figure 3 shows on the right an example of fragments extracted from an ornament. For several decades, hundreds of fragments coming from a large number of administrative documents have been extracted. This results in a complex puzzle in which some pieces are missing and most are very damaged. Papyrologists have

no choice but to resolve this complex puzzle before being able to analyse the writings. The amount of fragments to be analysed makes the assembly task beyond the scope of manual work only. They need to be assisted with a digital process for reconstructing the papyri. Methods already exist to solve puzzle-type problems. They mainly tackle the cases in which the documents are contemporary and all the pieces are available, which is not the case with papyrus fragments. In addition, most of them are hand-crafted methods. A strong knowledge of the content of the image is necessary in order to be able to adjust all the parameters, making these methods difficult to generalize.

In this article, we describe the first version of a tool that aims to help the Papyrologists to sort the fragments according to an image similarity criterion. Our main contribution is the creation of a process providing a list of similar fragments for a requested fragment. For that purpose, we tuned a deep siamese neural network architecture. As presented in figure 1[1], a papyrus can be composed of many fragments on which we extract patches. Figure 2 describes how our network is trained. We extract several patches from each fragment. We train the network on carefully selected pairs of patches belonging to the same papyrus and patches of fragments coming from different papyri.

2 RELATED WORK

The problem addressed in this work is a first step towards solving lots of puzzles of which all the pieces are mixed together. It is first necessary to identify which pieces come together in the same puzzle before tackling the assembly process. In the context of our work, we have multiple papyri (our puzzles) that have been torn apart in several fragments (our pieces) that are all mixed up together. The puzzle problem has mainly been studied in the context of natural images. This objective of reconstructing an image from many fragments is addressed as a np-hard optimization problem where, like in [17], pieces of fragments of photos are matched then arranged using shape and image content information. Recently, methods based on deep neural networks [20] open the way to other approaches.

In the specific context of document image reconstruction, some researchers have been interested in torn newspapers [19] but most focused on the reconstruction of typewritten documents shredded by a machine [3]. Indeed, in the past many documents (military archives, intelligence services) have been shredded to make information disappear. These fragments were digitized for historical purposes and must be reassembled so that historians can analyze them. This research area was very active in the early 2010's, a competition has even been initiated. It has been created by the U.S. Defence Advanced Research Projects Agency (DARPA) [6].

The puzzle problem can be addressed in different ways. One is to consider it as an optimization problem. In [7, 11], the authors propose relatively similar approaches to automatically resolve the puzzle. They first extract and approximate the contours. As they are in a context where all the fragments are available with contours that perfectly match from one fragment to another, they propose to model the problem as a cost function to minimize, like a similarity measure between puzzle pieces contours.

[1]Image taken from the University of Michigan Library Digital Collections

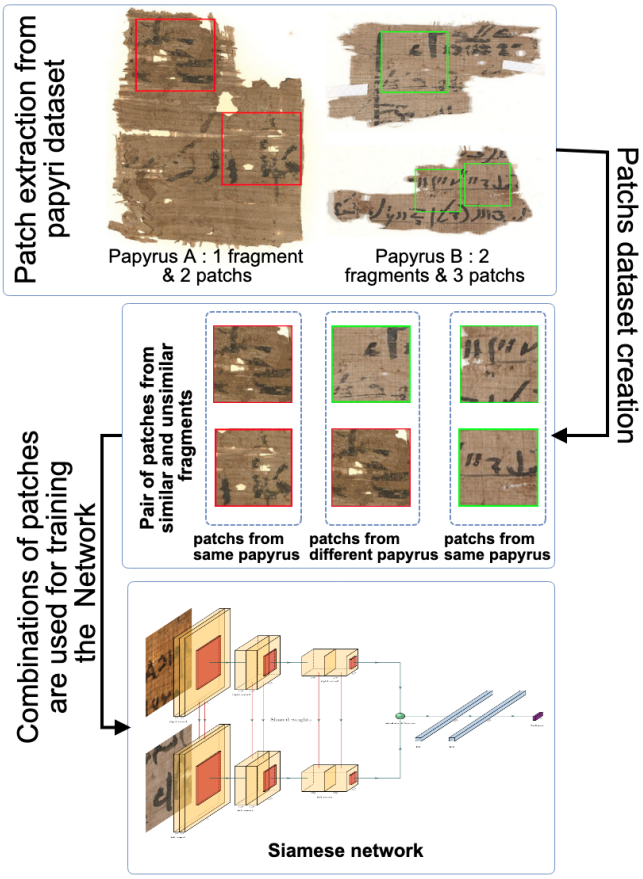


Figure 2: Overview of Papy-S-Net. The network is trained with similar and dissimilar pairs as input. In our case, the pairs are patches extracted from papyrus fragments. The network decides if 2 patches belong to the same fragment.

Figure 3: Left image : a funeral mask made from papyrus fragments. Right image : several fragments extracted from a decoration

Another way is to apply greedy methods to solve the puzzle. In [18], features are computed on contours of each fragment : contour length, color histogram ... these features are used to train a SVM (Support-Vector Machine) to find matching pairs of points of adjacent fragments. The alignment between fragments is done using a graphical modeling (vertex=fragment, edge=alignment) by trying

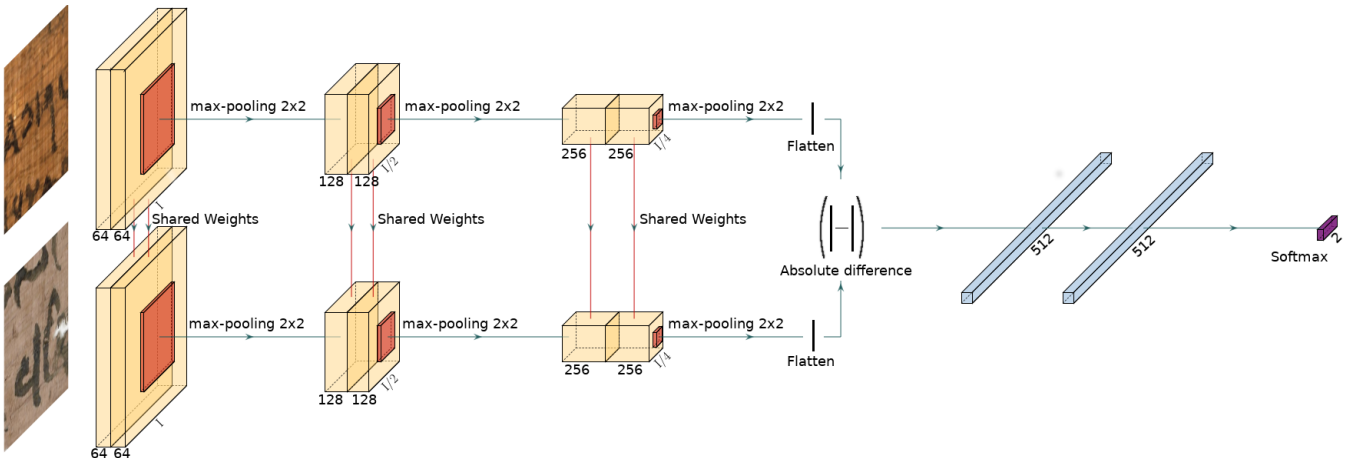


Figure 4: The architecture of Papy-S-Net. The input images are of size 128x128x3, each pooling operation (red squares) divides the dimension of the images by 2, each convolution operation uses a kernel of size 3x3. We use *ReLU* activation functions everywhere except for the last layer, which uses *Softmax* for binary classification. The weights of the two branches are shared and their outputs are combined by taking the absolute value of the element wise difference of the flattened outputs of each branch. This vector is then given to a *Softmax* two layers fully-connected network.

to find a spanning tree in it. The greedy methods assume that the dataset size remains reasonable.

As far as we know, most of computer science papers dealing with papyrus image analysis like [2, 9] aim to improve the digitalization process of physical documents and then to carry out image processing algorithm in order to be able to clearly distinguish the ink from the letters and the background pixels. Only few recent papers like [16] detail methods which are going further than just enhancing the image readability. Indeed, they present a deep architecture to precisely segment the fragment without being disturbed by the presence of the ruler, color charts and the scotch tape that hold the fragment on the support. They also propose to use a classical image retrieval process where features like : the size, the shape and SIFT descriptors are used to compare the fragments with each other. This fragment comparison allows to find the exact copies of previously digitized fragments (with a different acquisition protocol).

3 A DEEP SIAMESE NEURAL NETWORK TO MATCH PAPYRUS FRAGMENTS

3.1 Siamese architectures

Siamese networks are a special kind of neural networks having the particularity to contain two (or more) identical sub-networks (sometimes called branches). In a siamese network the layers arrangement is the same across the sub-networks, all their parameters are constrained to the same values. The goal is to define a similarity metric to decide if the inputs are similar or not. Siamese networks originally have been designed to compare handwritten signatures [5] with great success. More recently, they have been getting increasingly popular with applications in face recognition [21], object tracking [4] or analysis of medical images [8].

More recently, siamese networks have been used by Kassis [12] to align portions of the same text coming from two different documents and to find matching words pairs. Zhong et al. [23] use siamese and pseudo-siamese networks to perform word spotting in handwritten historical documents. We can also note [1] which uses *autoencoders* together with siamese networks for automatic signature verification.

3.2 Architecture of Papy-S-Net

The proposed siamese architecture was inspired by the architecture presented by Koch et al. in [14], where a deep siamese network is used for one-shot learning of handwritten characters from a variety of languages (see the Omniglot dataset, introduced in [15]). This architecture is designed to work on dataset which's structure resembles the Omniglot dataset images, that is black on white images containing characters. Our dataset is very different. Although also containing characters, they are written on papyri, which contain a lot of texture information (see Figure 2) that can be exploited for better recognition. We describe here the architecture of **Papy-S-Net** and how and why it differs from Koch et al.'s.

Papy-S-Net (a diagram of the architecture can be seen on Figure 4[2]) consists of two identical convolution / pooling branches with shared weights, meaning that at any time, every corresponding neuron weights are the same between the two branches. Contrary to Koch et al.'s architecture, we choose to have two consecutive convolutional layers before applying the max pooling operation every time. Indeed, each pooling operation reduces the size of the image (by a factor of 2 in our case), which reduces the amount of details available. Thus, multiplying sequential convolution operations before pooling allows for a finer representation of the data at each scale. This is useful to capture local details, such as texture, as well as global information on the image, such as writing style.

After flattening the outputs of the convolutional parts of the network, we compute the element-wise absolute difference of the two feature vectors. Some siamese networks perform a concatenation of these vectors instead of this absolute difference (i.e. [22, 23]), but trying it out, we noticed it created an asymmetry in the results :

[2]This diagram was generated using github.com/HarisIqbal88/PlotNeuralNet

permuting the input branches on the same pair of patches outputted different results.

As we do not really want a similarity score as in most siamese networks, but rather a 2-by-2 classification of which patches come from which fragments, we choose the output of **Papy-S-Net** to be a 2 classes *Softmax* function ([dissimilar, similar]) after the two fully-connected layers. This, combined with the use of the *cross-entropy* loss function, tends to push the classification scores to extremes (close to 0 and close to 1) which makes the decision process easy (no need for an expertly tuned threshold), but removes the possibility of manually adjusting the threshold to filter false positives.

For training, we use a batch size of 64, a learning rate of 0.0001 and the *AdamOptimizer* [13] *Tensorflow* implementation. We train for 80 epochs as accuracy and loss values stabilize around this number and the performance do not improve by training for longer. At each epoch during training, the batches are recomputed, randomizing the order in which the network sees all the positive and negative pairs available in the training set. The validation set is fixed at the beginning of training, and the validation accuracy is computed on all the set at the end of each epoch. All the experiments presented in the next sections are done on an Nvidia Titan X (Pascal) with 12GB of video memory and using *Tensorflow* version 1.13.

3.3 Dataset creation process

In the context of deep learning, a large dataset is needed for correctly training a model. A model is considered good if it can generalize well to new data it has never seen before. It means that it has learned the underlying concept represented in the data it was trained on, and did not just memorize the data, which would be *Overfitting*.

Our dataset consists of just over 500 papyrus fragments (from now on, this dataset will be called *B500*). The papyri are from the University of Michigan Library Digital Collections. These papyri have very varied contents since they come from different periods (from 600 years BCE to 400 years CE), they have been written in different languages (Coptic, Greek, Arabic, Hebrew, Hieratic, Latin and Demotic) and are of different sizes. They have also been digitized at different times with different settings. Because real ground truth dataset is very difficult to obtain in our context, we choose to create our ground truth automatically by extracting patches from the fragments. As the fragments differ in size, we obtain a different number of patches for each of them. In order to be accepted as part of the dataset, a patch needs to contain at least 80% of *material* (i.e. less than 20% of background). This approach allows us to generate enough ground truth to train the deep siamese neural network. The goal of the network is now to decide if two different patches come from the same fragment. In order to test if the results are influenced by the presence or absence of text in the patches, we conduct the experiments with three sets created with different patch extraction approaches : a set of patches all containing text, a set of patches containing only texture (no text) and a set with randomly selected patches (no constrains on the presence or not of text).

In order to extract only patches containing text or patches not containing text, we need a way to locate the lines of text in the papyrus fragments. We used the first part of **ARU-Net** [10] to do so. Results can be seen in Figure 5.

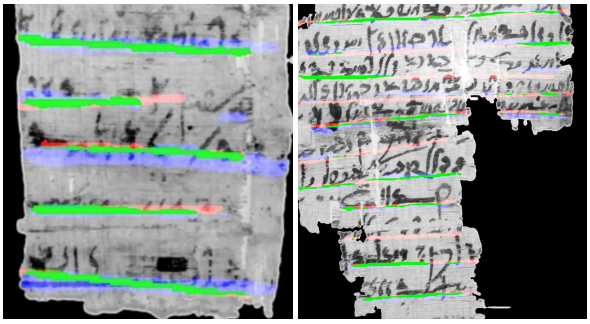


Figure 5: Typical line detection results obtained with ARU-Net on our dataset. Green, red and blue pixels respectively denote correct, false positives and false negatives relative to manually labeled ground truth.

With this information, we can now extract three different kinds of (non overlapping) patches on which to perform the experiments. *Random* patches are extracted without any constraints. *Without text* patches are selected so that they *do not* contain text, only background (papyrus texture). *With text* patches are selected so that they all *do* contain text. All the extracted patches do not contain borders because we do not want to learn the shape of the fragments but their contents. We extract as many non overlapping patches which satisfy their respective constraints as possible for each fragment. Naturally, we get a lot more of *Random* patches than of *Without text* or *With text*. To be able to compare the results, we need to have roughly the same amount of data for the three approaches. We therefore randomly discard patches in the *Random* class until getting the same number of patches in the three classes. It is important to note that the number of patches extracted per fragment can vary quite a lot depending on the size of the given fragment, ranging from 2, for the smallest ones, to 123 for the biggest ones.

We then prepare our datasets by following the classic `Train / Validation / Test` split. We first split the dataset into [`Train / Validation`], 90% and [`Test`], 10%. We get about 1000 patches coming from about 50 fragments for the Test set. The [`Train / Validation`] part is then split in `Train`, 80% and `Validation`, 20%, giving us about 8500 patches for the `Train` set and about 2000 patches for the `Validation` set. For these sets, we create the same number of *similar* (patches coming from the same fragment) and *dissimilar* (patches coming from different fragments) pairs. Batches for training are pre-computed at each new epoch, we ensure that all examples of the `Train` set are seen at each epoch in a random order by associating two patches for each patch in the set, one *similar* and one *dissimilar*. We also add the permutation of each created pair to the dataset. Finally, statistics on the Test set are computed on all the possible pairs permutations on the set.

4 EXPERIMENTS AND RESULTS

We compare the results obtained on **Papy-S-Net** and Koch et al.'s architectures with our three patches extraction approaches on two test datasets. As can be seen on Figure 9, the best accuracy performance is achieved with our method on patches containing text after 70 epochs. This seems to be confirmed by Figures 6 and 7.

It is interesting to note that we respectively obtain globally better results when trained with *Random* patches than with *Without text* patches, our model being slightly better or very close most of the time. However, we can notice an improvement on **Papy-S-Net** when trained on patches *With text*. As can be seen on Figures 6 and 7, **Papy-S-Net** outperforms Koch et al.'s for all metrics in this case, and actually almost outperforms every other approaches (see bold numbers). We tested the different approaches on two datasets.

	Random		Without Text		With Text	
Rates	PS-Net	Koch	PS-Net	Koch	PS-Net	Koch
True Pos.	0.80	0.74	0.75	0.76	**0.82**	0.72
True Neg.	0.92	0.88	0.82	0.87	**0.94**	0.86
False Pos.	0.09	0.12	0.08	0.13	**0.06**	0.14
False Neg.	0.20	0.26	0.25	0.24	**0.18**	0.28

Figure 6: Comparing the performance on *B500* obtained for our architecture and that of Koch, et al. The best-performing method for each metric is listed in bold-faced type, and the worst method is underlined

The first one is the Test part of each training for each patch extraction method, as explained in section 3.3. We can see on Figure 6 that the best True positive, False Positive, True Negative and False Negative rates are obtained when training **Papy-S-Net** on patches *With text*, and most of the worst rates are obtained when training Koch et al.'s architecture on patches *With text*. While the performance difference between these two examples is quite significant (between 8 and 10 percentage points), the respective performance differences between the *Random* and *With text* approaches exist, but is smaller. Taking into account the overall pre-processing cost to detect the baselines in order to extract patches all containing text, which in itself can probably be improved and brings its own uncertainties, the *Random* approach seems to bring acceptable results. However, random being what it is, this approach may not cope well with papyri containing little text, as it would statistically tend to extract lots of patches only containing texture, which is the worst performing approach with our model. In this case, we would recommend the *With text* approach.

	Random		Without Text		With Text	
Rates	PS-Net	Koch	PS-Net	Koch	PS-Net	Koch
True Pos.	0.77	0.64	0.77	0.74	**0.79**	0.75
True Neg.	0.81	0.71	**0.87**	0.82	0.83	0.74
False Pos.	0.19	0.29	**0.13**	0.18	0.17	0.26
False Neg.	0.23	0.23	0.23	0.26	**0.21**	0.25

Figure 7: Comparing the performance on *RealUseCase* obtained for our architecture and that of Koch, et al. The best-performing method for each metric is listed in bold-faced type, and the worst method is underlined.

The second one is a subset of the aforementioned Test part on which we know that different fragments actually belong to the same papyrus, let's call it *RealUseCase*. This dataset contains 60 fragments belonging to 19 papyri. This is closer to a real world case, where we have different fragments and we try to find which belong to the same papyrus. The *B500* dataset was used to train the model, this dataset is emulating the real work of Papyrologists. Knowing the ground truth information of the correct belongings of the fragments, we can see if patches are correctly classified in their corresponding papyrus, which is composed of two or more fragments. Again, as can be seen on Figure 7, **Papy-S-Net** outperforms Koch et al.'s architecture in nearly all experiments.

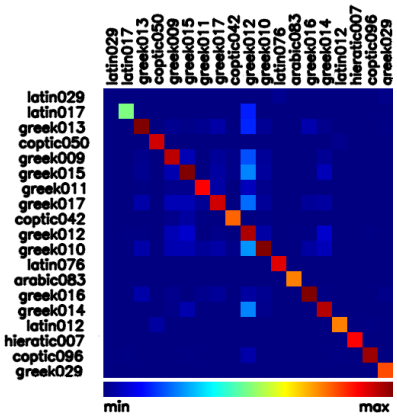


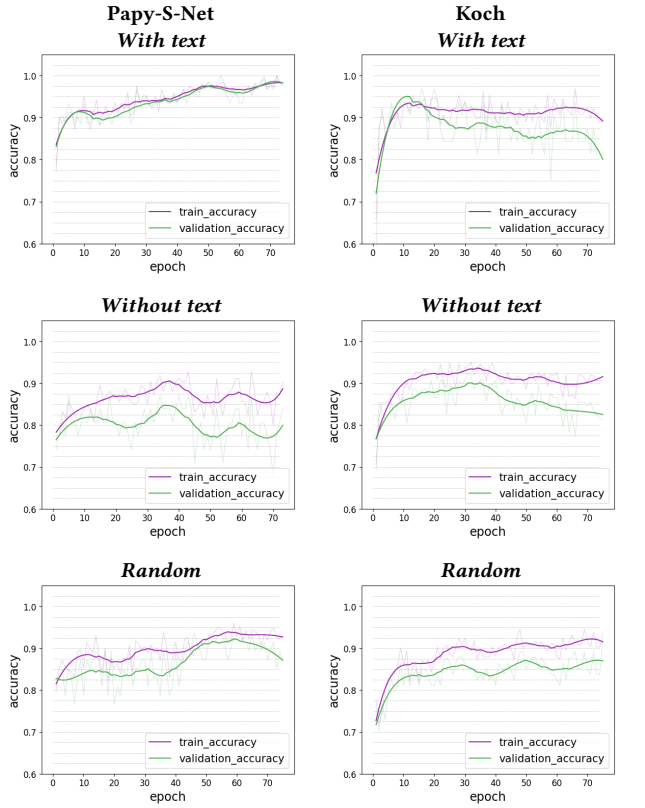
Figure 8: Confusion matrix on the *RealUseCase* dataset for the *With text* patch extraction approach. The names (such as "greek016") represent a papyrus composed of at least two different fragments.

Figure 8 shows correct and incorrect pairings on the *RealUseCase* dataset in the form of a confusion matrix. For each papyrus, we compare the extracted patches to the other patches of all other papyri and register which pairs were considered as "similar" and increment the number in the corresponding case of the matrix, which is afterwards normalized. We can see on the diagonal that the patches are mostly classified in the correct papyri. Interestingly, the papyrus "greek012" seems to attract a lot of false positives, especially on other papyrus containing greek language. One reason for that could be because 96 patches were extracted on this papyrus, which is a lot more than most of the others, making this class statistically more likely to be classified incorrectly. Another reason could be that the network implicitly learns some language classification skills.

5 CONCLUSION AND FUTURE WORK

With the introduction of **Papy-S-Net**, we have proposed a promising method for sorting papyri fragments in terms of their belonging to the same papyrus. Along with the siamese network itself, we proposed a patch based methodology in order to prepare the datasets, and concluded that training the network using patches all containing text yielded the best matching results. The pre-processing phase needed for this approach can be very much improved, for example, by accurately segmenting the papyri, we could be more precise in selecting patches without background.

Papy-S-Net is a good first step towards the final goal of our research, which is actually reconstructing the puzzle. It can already be used to perform a significant sorting process on existing fragments databases, however we still suffer from a too high false negative rate for the needs of the Papyrologists, it is indeed worst to incorrectly discard fragments than to accept wrong fragments which could



	Random		Without Text		With Text	
	PS-Net	Koch	PS-Net	Koch	PS-Net	Koch
Train acc.	0.93	0.91	0.89	0.92	**0.97**	0.92
Val. acc.	0.90	0.86	0.84	0.89	**0.96**	0.87

Figure 9: Train and Validation accuracy plots and table for the different experiments

be further filtered by other means. This issue could be addressed by a more linear output of the network, giving the ability to express doubt about the result instead of classifying binarily. We also want to continue this work by experimenting with patch level data augmentation which, combined with better pre-processing, may greatly improve the current results.

ACKNOWLEDGMENTS

The research leading to this results has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 758907 and is part of the *GESHAEM* project, hosted by the *Ausonius* Institute. The tests were carried out on images from the Image Bank, University of Michigan Library Digital Collections. https://quod.lib.umich.edu/a/apis Accessed: June 04, 2019. we also thank Pierre Biasutti for his advice and Pierre Célor for his work during his master's internship.

REFERENCES

[1] Kian Ahrabian and Bagher BabaAli. 2017. On Usage of Autoencoders and Siamese Networks for Online Handwritten Signature Verification. *CoRR* abs/1712.02781 (2017). arXiv:1712.02781

[2] Athina A Alexopoulou, Agathi-Anthoula Kaminari, Athanasios Panagopoulos, and Egert Pöhlmann. 2013. Multispectral documentation and image processing analysis of the papyrus of tomb II at Daphne, Greece. *Journal of Archaeological Science* 40, 2 (2013), 1242–1249.

[3] Ahmed S Atallah, E Emary, and Mohamed S El-Mahallawy. 2015. A step toward speeding up cross-cut shredded document reconstruction. In *2015 Fifth International Conference on Communication Systems and Network Technologies*. IEEE, 345–349.

[4] Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. 2016. Fully-Convolutional Siamese Networks for Object Tracking. *CoRR* abs/1606.09549 (2016). arXiv:1606.09549

[5] J. Bromley, I. Guyon, Y. LeCun, and R. Shah E. Sackinger. 1994. Signature verification using a "siamese" time delay neural network. *NIPS* (1994), 737âĂŞ744.

[6] Patrick Butler, Prithwish Chakraborty, and Naren Ramakrishan. 2012. The deshredder: A visual analytic approach to reconstructing shredded documents. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 113–122.

[7] Junhua Chen, Miao Tian, Xingming Qi, Wenxing Wang, and Youjun Liu. 2019. A Solution to Reconstruct Cross-Cut Shredded Text Documents Based on Constrained Seed K-means Algorithm and Ant Colony Algorithm. *Expert Systems with Applications* (2019).

[8] Nikhil Bhagwat et al. 2018. Modeling and prediction of clinical symptom trajectories in AlzheimerâĂŹs disease using longitudinal data. *PLOS, Computational Biology* 14, 9 (2018), 1–25.

[9] Marco Gargano, Duilio Bertani, Marinella Greco, John Cupitt, Davide Gadia, and Alessandro Rizzi. 2015. A perceptual approach to the fusion of visible and NIR images in the examination of ancient documents. *Journal of Cultural Heritage* 16, 4 (2015), 518–525.

[10] Tobias Grüning, Gundram Leifert, Tobias Strauß, and Roger Labahn. 2018. A two-stage method for text line detection in historical documents. *arXiv preprint arXiv:1802.03345* (2018).

[11] Edson Justino, Luiz S Oliveira, and Cinthia Freitas. 2006. Reconstructing shredded documents through feature matching. *Forensic science international* 160, 2-3 (2006), 140–147.

[12] M. Kassis, J. Nassour, and J. El-Sana. 2017. Alignment of Historical Handwritten Manuscripts Using Siamese Neural Network. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 01. 293–298. https://doi.org/10.1109/ICDAR.2017.56

[13] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980 cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[14] Gregory R. Koch. 2015. Siamese Neural Networks for One-Shot Image Recognition.

[15] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science* 350, 6266 (2015), 1332–1338. https://doi.org/10.1126/science.aab3050

[16] Gil Levi, Pinhas Nisnevich, Adiel Ben-Shalom, Nachum Dershowitz, and Lior Wolf. 2018. A Method for Segmentation, Matching and Alignment of Dead Sea Scrolls. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 208–217.

[17] Hairong Liu, Shengjiao Cao, and Shuicheng Yan. 2011. Automated assembly of shredded pieces from multiple photos. *IEEE Transactions on Multimedia* 13, 5 (2011), 1154–1162.

[18] Fabian Richter, Christian X Ries, Nicolas Cebron, and Rainer Lienhart. 2012. Learning to reassemble shredded documents. *IEEE Transactions on multimedia* 15, 3 (2012), 582–593.

[19] Fabian Richter, Christian X Ries, Stefan Romberg, and Rainer Lienhart. 2014. Partial contour matching for document pieces with content-based prior. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

[20] Dror Sholomon, Omid E David, and Nathan S Netanyahu. 2016. DNN-Buddies: A deep neural network-based estimation metric for the jigsaw puzzle problem. In *International Conference on Artificial Neural Networks*. Springer, 170–178.

[21] Haoran Wu, Zhiyong Xu, Jianlin Zhang, Wei Yan, and Xiao Ma. 2017. Face Recognition based on Convolution Siamese Networks. In *10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI 2017)*.

[22] Sergey Zagoruyko and Nikos Komodakis. 2015. Learning to Compare Image Patches via Convolutional Neural Networks. *CoRR* abs/1504.03641 (2015). arXiv:1504.03641

[23] Z. Zhong, W. Pan, L. Jin, H. Mouchere, and C. Viard-Gaudin. 2016. SpottingNet: Learning the Similarity of Word Images with Convolutional Neural Network for Word Spotting in Handwritten Historical Documents. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 295–300. https://doi.org/10.1109/ICFHR.2016.0063