Kraken - an Universal Text Recognizer for the Humanities

I. Introduction

Retrodigitization of both printed and handwritten material is a common prerequisite for a diverse range of research questions in the humanities. While optical character recognition on printed texts is widely considered to be fundamentally solved in academia, with the most commonly used paradigm [1] dating back to 2006, this hasn't translated into increased availability of adaptable, libre-licensed OCR engines to the technically inclined humanities scholar.

The nature of the material of interest commands a platform that can be altered with minimum effort to achieve optimal recognition accuracy; uncommon scripts, historical languages, complex or archaic page layout, and non-paper writing surfaces are rarily satisfactorily addressed by off-the-shelf commercial solutions. In addition, an open system ameliorates the severe resource constraints of humanities research by enabling sharing of artifacts, such as training data and recognition models, inaccessible with proprietary OCR technology.

II. Kraken

The Kraken text recognition engine is an extensively rewritten fork of the OCRopus system. It can be used both for handwriting and printed text recognition, is easily (re-)trainable, and great care has been taken to eliminate implicit assumptions on content and layout that complicate the processing of non-Latin and non-modern works.

Thus Kraken has been extended with features and interfaces enabling the processing of most scripts, among them full Unicode right-to-left, bidirectional, and vertical writing support, script detection, and multiscript recognition. Processing of scripts not included in Unicode is also possible through a simple JSON interface to the codec mapping numerical model outputs to characters. The same interface provides facilities for efficient recognition of large logographic scripts.

Output includes fine-grained bounding boxes down to the character level that may be used to quickly acquire a large number of samples from a corpus to assist in paleographic research. Kraken implements a flexible output serialization scheme utilizing a simple templating language. Templates are available for the most commonly used formats ALTO, hOCR, TEI, and abbyyXML.

While including implementations of all the subprocesses needed in a text recognition pipeline, most functional blocks can be accessed separately on the command line, allowing flexible substitution of specially optimized methods. A stable programming interface allows total customization and integration into other software packages.

A. Recognition

The recognition engine operates as a segmentation-less sequence classifier using an artificial neural network to map an image of a single line of text, the input sequence, into a sequence of characters, the output sequence. The artificial neural network employed is a combination convolutional and recurrent neural network trained with the CTC loss function [1] that reduces training data requirements to line-level transcriptions (figure 2). Regularization is mainly provided by dropout [2] after both convolutional and recurrent layers. User intervention in determining training duration and model selection is largely eliminated through early stopping.

Specialized networks, e.g. for particularly complex scripts, can be assembled from building blocks with a simple network specification language although the default architecture shown in figure 1 is suitable for the vast majority of applications.

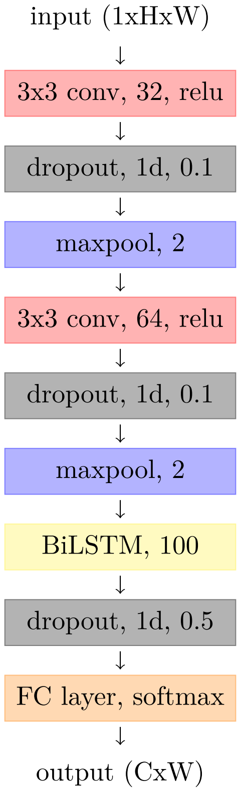


Fig. 1: Network architecture (H: sequence height, W: sequence length, C: alphabet size)

Processing of dictionaries and library catalogues with extensive semantic markup such as italic, underlining, and bolding, is also possible through specially prepared training data.

B. Layout Analysis and Script Detection

Kraken's layout analysis extracts text lines from an input image for later processing by the recognition engine.

Apart from a basic segmenter taken from OCRopus a trainable line extractor is in the process of being implemented. Full trainability of layout analysis is of utmost importance to a truly universal OCR system, as text layout and its semantics varies widely across time and space, e.g. hand-crafted methods for printed Latin text are unlikely to work reliably on Arabic text or manuscripts with extensive interlinear annotation.

The layout analysis module consists of a two-step instance segmentation method: an initial seed-labelling network operates on the whole page labelling the area between baseline and mean of each line. These easily separable line seeds are fed with the surrounding region of interest into a second smaller network that expands seeds to whole line masks. In contrast to the semantic segmentation methods common in neural layout analysis this approach does not require postprocessing to extract lines from pixel labellings. It is therefore intrinsically capable of finding arbitrarily oriented or distorted lines which is generally not true for other layout analysis tools.

	Mean character accuracy	**Standard deviation**	**Maximum accuracy**
Prints			
Arabic[a]	99.5%	0.05	99.6%
Persian[b]	98.3%	0.33	98.7%
Syriac[c]	98.7%	0.38	99.2%
Polytonic Greek[d]	99.2%	0.26	99.6%
Latin[e]	98.8%	0.09	99.3%
Latin incunabula[f]	99.0%	0.11	99.2%
Fraktur[g]	99.0%	0.31	99.3%
Cyrillic[h]	99.3%	0.15	99.6%
Manuscripts			
Hebrew[i]	96.9%	-	-
Medieval Latin[j]	98.2%	-	-

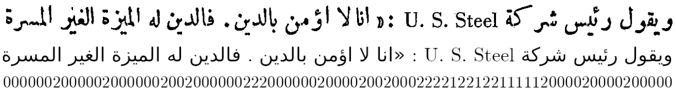
[a] Mid-20th century vocalised Arabic
[b] Mid-20th century printing
[c] Late-19th century printing in Serṭā form
[d] Late-19th century printing
[e] 12 prints ranging from 1471 to 1686
[f] 3 prints (1471 to 1483)
[g] 20 prints (1487 to 1870)
[h] 1923 Russian print
[i] Medieval Midrash Tanhuma
[j] Mid-9th century Carolingian of Josephus Latinus

TABLE I: Mean character accuracy and standard deviation on the validation set across 10 training runs on each training set

The seed-labelling network is a modified U-net [3] on the basis of a 34-layer residual network [4] pretrained on ImageNet. Second-stage expansion is through a three layer CNN. Both are trained on a training data set of labelled line segmentations.

Preliminary results on a semi-automatically generated data set of modern English text can be seen in figure 4.

Script detection, the basis for multi-script support in the recognizer, is implemented as a segmentation-less sequence classification problem, similar to text recognition. Instead of assigning a unique label to each code point or grapheme cluster we assign all code points of a particular script the same label. The network is then trained to output the correct sequence of script labels (figure 2). The output sequence is then used to split the line into single-script runs that can be classified with monolingual recognition models (figure 3).



ويقول رئيس شركة U. S. Steel : «انا لا اؤمن بالدين . فالدين له الميزة الغير المسرة

000000200000200000020020000002220000000200002002000222212212211111200000200002000000

Fig. 2: Original and modified ground truth (top: original line, middle: transcription,
bottom: assigned script classes)

III. Results

kraken has been used on a wide variety of writing systems, achieving uniformly high character accuracy (CER). Sample accuracies for a diverse set of scripts spanning across multiple centuries of printing are shown in table I.

As a special use case we evaluated recognition of text and emphasis in a mixed English and romanized Arabic library catalog on a training set of 350 lines (50 lines in the validation set) resulting in an averaged CER of 99.3% ($\sigma = 0.16$) over 10 runs with 95.38% CER on cursive and text with increased spacing ($\sigma = 1.46$). When using only emphasized text accuracy as the stopping criterium mean accuracy rises to 99.03% ($\sigma = 0.28$).

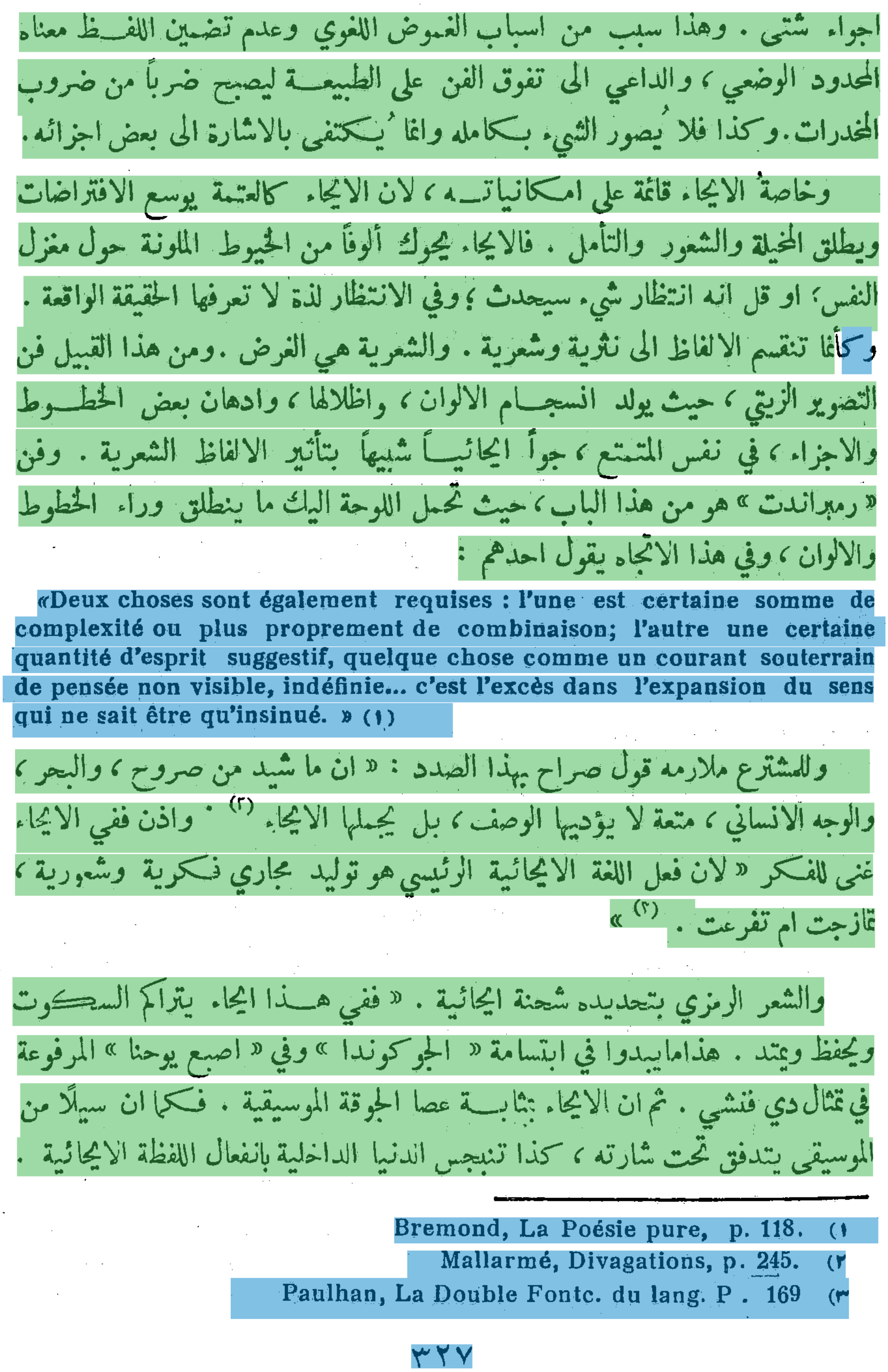
References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, ACM, 2006, pp. 369–376.

[2] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.



اجواء شتى . وهذا سبب من اسباب الغموض اللغوي وعدم تضمين اللفـظ معناه
المحدود الوضعي ، والداعي الى تفوق الفن على الطبيعـة ليصبح ضرباً من ضروب
المخدرات. وكذا فلا يُصور الشيء بكامله وانما يُكتفى بالاشارة الى بعض اجزائه.

وخاصةُ الايحاء قائمة على امكانياتـه ، لان الايحاء كالعتمة يوسع الافتراضات
ويطلق المخيلة والشعور والتأمل . فالايحاء يحوك ألوفاً من الخيوط الملونة حول مغزل
النفس؛ او قل انه انتظار شيء سيحدث ؛ وفي الانتظار لذة لا تعرفها الحقيقة الواقعة .
وكأنما تنقسم الالفاظ الى نثرية وشعرية . والشعرية هي الغرض . ومن هذا القبيل فن
التصوير الزيتي ، حيث يولد انسجـام الالوان ، واظلالها ، وادهان بعض الخطـوط
والاجزاء ، في نفس المتمتع ، جواً ايحائيـاً شبيهاً بتأثير الالفاظ الشعرية . وفن
« رمبراندت » هو من هذا الباب ، حيث تحمل اللوحة اليك ما ينطلق وراء الخطوط
والالوان ، وفي هذا الاتجاه يقول احدهم :

«Deux choses sont également requises : l'une est certaine somme de complexité ou plus proprement de combinaison; l'autre une certaine quantité d'esprit suggestif, quelque chose comme un courant souterrain de pensée non visible, indéfinie... c'est l'excès dans l'expansion du sens qui ne sait être qu'insinué. » (١)

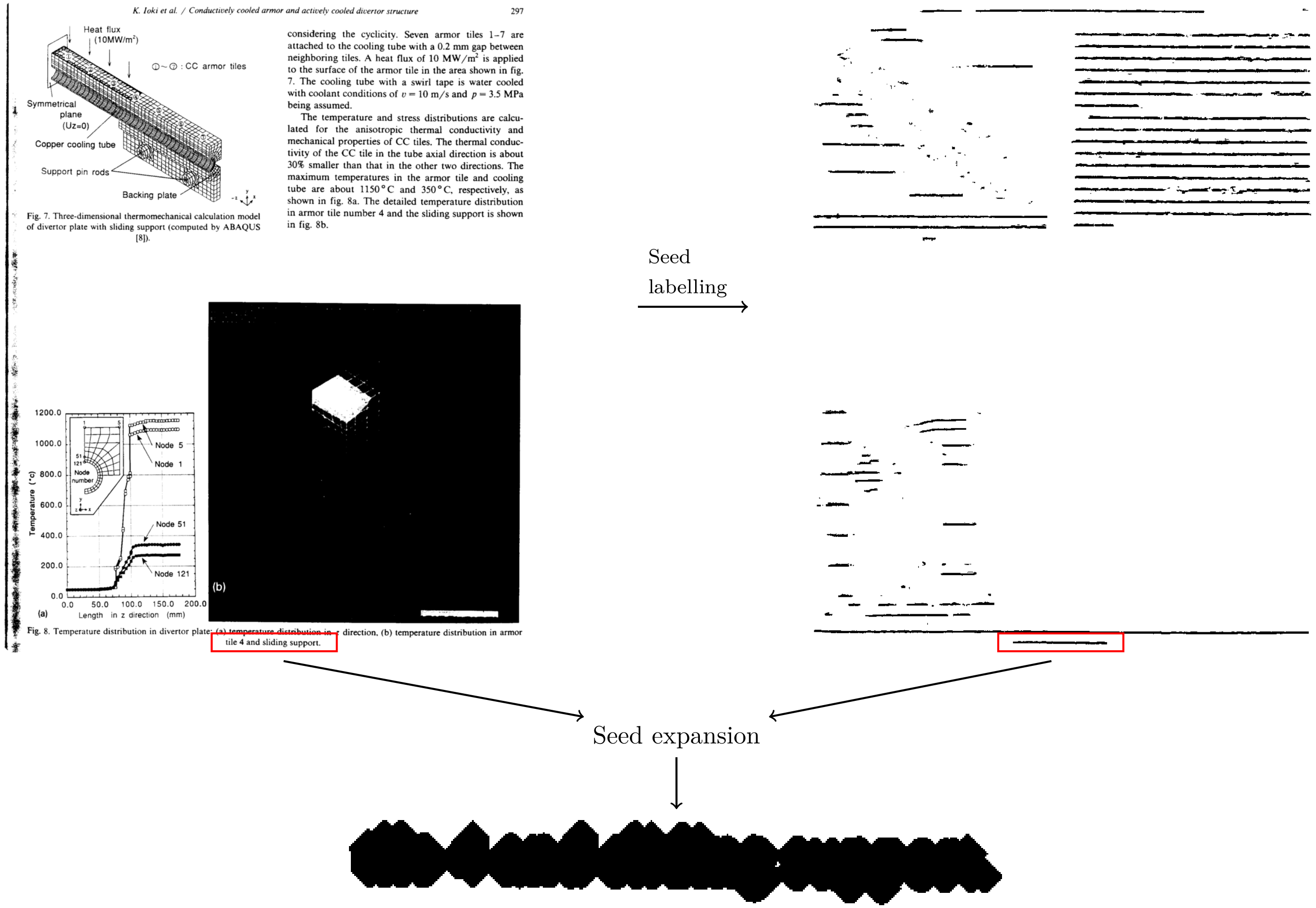
وللمشترع ملارمه قول صراح بهذا الصدد : « ان ما شيد من صروح ، والبحر ،
والوجه الانساني ، متعة لا يؤديها الوصف ، بل يحملها الايحاء (٢) . واذن ففي الايحاء
غنى للفكر « لان فعل اللغة الايحائية الرئيسي هو توليد مجاري فكرية وشعورية ،
تمازجت ام تفرعت . (٢) »

والشعر الرمزي بتحديده شحنة ايحائية . « ففي هـذا ايحاء يتراكم السكوت
ويحفظ ويمتد . هذامايبدوا في ابتسامة « الجوكوندا » وفي « اصبع يوحنا » المرفوعة
في تمثال دي فنشي . ثم ان الايحاء بمثابـة عصا الجوقة الموسيقية . فكما ان سيلاً من
الموسيقى يتدفق تحت شارته ، كذا تنبجس الدنيا الداخلية بانفعال اللفظة الايحائية .

Bremond, La Poésie pure, p. 118. (١
Mallarmé, Divagations, p. 245. (٢
Paulhan, La Double Fontc. du lang. P . 169 (٣

٣٢٧

Fig. 3: Sample output of the script detection on a bilingual French/Arabic page. Note that Eastern Arabic are always classified as Latin text



K. Ioki et al. / Conductively cooled armor and actively cooled divertor structure 297

Fig. 7. Three-dimensional thermomechanical calculation model of divertor plate with sliding support (computed by ABAQUS [8]).

considering the cyclicity. Seven armor tiles 1–7 are attached to the cooling tube with a 0.2 mm gap between neighboring tiles. A heat flux of 10 MW/m^2 is applied to the surface of the armor tile in the area shown in fig. 7. The cooling tube with a swirl tape is water cooled with coolant conditions of $v = 10$ m/s and $p = 3.5$ MPa being assumed.

The temperature and stress distributions are calculated for the anisotropic thermal conductivity and mechanical properties of CC tiles. The thermal conductivity of the CC tile in the tube axial direction is about 30% smaller than that in the other two directions. The maximum temperatures in the armor tile and cooling tube are about 1150 °C and 350 °C, respectively, as shown in fig. 8a. The detailed temperature distribution in armor tile number 4 and the sliding support is shown in fig. 8b.

Fig. 8. Temperature distribution in divertor plate: ~~(a) temperature distribution in z~~ direction, (b) temperature distribution in armor tile 4 and sliding support.

Fig. 4: Preliminary results from the two-stage baseline segmenter.