Efficient OCR for Building a Diverse Digital History

Jacob Carlson[1], Tom Bryan[2], Melissa Dell[1,2*]

[1]Department of Economics, Harvard University; Cambridge, MA, USA.
[2]National Bureau of Economic Research; Cambridge, MA, USA.
[*]Corresponding author: melissadell@fas.harvard.edu.

Abstract: Thousands of users consult digital archives daily, but the information they can access is unrepresentative of the diversity of documentary history. The sequence-to-sequence architecture typically used for optical character recognition (OCR) – which jointly learns a vision and language model - is poorly extensible to low-resource document collections, as learning a language-vision model requires extensive labeled sequences and compute. This study models OCR as a character level image retrieval problem, using a contrastively trained vision encoder. Because the model only learns characters' visual features, it is more sample efficient and extensible than existing architectures, enabling accurate OCR in settings where existing solutions fail. Crucially, the model opens new avenues for community engagement in making digital history more representative of documentary history.

A sample-efficient, extensible, open-source OCR

Digital texts are central to the study, dissemination, and preservation of human knowledge. Tens of thousands of users consult digital archives daily in Europe alone (*12*), yet billions of documents remain trapped in hard copy in libraries and archives around the world. These documents contain extremely diverse character sets, languages, fonts or handwriting, printing technologies, and artifacts from scanning and aging. Converting them into machine-readable data that can power indexing and search, computational textual analyses, and statistical analyses - and be more easily consumed by the public - requires highly extensible, accurate, efficient tools for optical character recognition (OCR).

Current OCR technology - developed largely for small-scale commercial applications in high resource languages - falls short of these requirements. OCR is typically modeled as a sequence-to-sequence (seq2seq) problem, with learned embeddings from a neural vision model taken as inputs to a learned neural language model. The seq2seq architecture is challenging to extend and customize to novel, lower resource settings (*22*), because training a vision-language model requires a vast collection of labeled image-text pairs and significant compute. This study shows that on printed Japanese documents from the 1950s, the best performing existing OCR mis-predicts over half of characters. Poor performance is widespread, spurring a large post-OCR error correction literature (*37, 40, 53*) and skewing digital history towards limited settings that are not representative of the diversity of documentary history.

This study develops a novel, open source OCR architecture, EffOCR (**Eff**icient**OCR**), designed for researchers and archives seeking a sample-efficient, customizable, scalable OCR solution for diverse documents. EffOCR combines the simplicity of early OCR systems, such as Tauschek's 1920s reading machine, with deep learning, bringing OCR back to its roots: the *optical* recognition of *characters*. Deep learning-based object detection methods are used to localize individual characters in the document image. Character recognition is modeled as an image retrieval problem, using a vision encoder contrastively trained on character crops.

EffOCR performs very accurately, even when using lightweight models designed for mobile phones that are cheap to train and deploy. Using documents that are fundamental to studying Japan's remarkable 20th century economic growth, the study shows EffOCR can provide a sample efficient, highly accurate OCR architecture for contexts where all current solutions fail. EffOCR's blend of accuracy and efficient runtime also makes it attractive for digitizing massive-scale collections in high resource languages, which the study illustrates with Library of Congress's collection of historical U.S. newspapers (*34*).

Historically, the low accuracy of traditional computer vision methods made modeling language necessary for OCR, but recent revolutionary advances in computer vision obviate this need as long as the document is legible. This greatly reduces the cost of deploying OCR to novel settings, as EffOCR only needs to learn the visual features of characters, not how they are combined to form language. Extending EffOCR to novel settings requires just dozens to hundreds of labeled text line images, simple training recipes, and modest compute. In contrast to seq2seq, new characters can be added at inference time, important for contexts such as archaeology where new characters are regularly discovered. EffOCR furthers the mission of Layout Parser (*46*) - our widely used Python package for detecting document layouts - to make digital history more representative of documentary history.

Methods Overview

Modern OCR overwhelmingly uses deep neural networks - either a convolutional neural network (CNN) or vision transformer (ViT) - to encode images. The representations created by passing an input image through a neural encoder are then decoded to the associated text.

Figure 1 underscores two fundamental differences between EffOCR and seq2seq. First, sequence-to-sequence architectures typically require line level inputs, and individual characters are not localized; rather, images or their representations are divided into fixed size patches. In contrast, EffOCR localizes characters using modern object detection methods (*5, 27*). Second, seq2seq sequentially decodes the learned image representations into text using a learned language model that takes the image representations as inputs. In contrast, EffOCR recognizes text by using contrastive training (*30*) to learn a meaningful metric space for character-level OCR. The vision encoder projects crops of the same character - regardless of style - nearby, whereas crops of different characters are projected further apart. Character embeddings are decoded to text in parallel by retrieving their nearest neighbor in an offline index of exemplar embeddings, created by rendering character images with a digital font.

Different vision encoders can be used interchangeably for the EffOCR character localizer and recognizer. Three models are considered: a vision transformer (*3, 5*) model (EffOCR-T Base), a convolutional (*5, 36*) base model (EffOCR-C Base), and a convolutional small model (EffOCR-C Small), which uses lightweight architectures designed for mobile phones (*23, 27*). The supplementary materials further describe the EffOCR architecture and training recipes and evaluate additional models.

Evaluation Datasets

Evaluating EffOCR requires benchmark datasets that are representative of the diversity of documentary history. Traditional OCR benchmarks focus on commercial applications like receipts (*24*), so this study builds novel datasets. First, the study draws on documents from historical Japan that can elucidate fundamental questions that have been understudied due to a lack of digital data, such as the drivers of Japan's rapid transformation from a poor agrarian economy to a wealthy industrialized nation. Horizontally and vertically written tabular data - providing rich information on Japanese firms and their personnel - are drawn from two 1950s publications (*26, 51*). A 1930s prose publication providing detailed biographies of tens of thousands of individuals (*25*) is also examined. These texts could use over 13,000 *kanji* characters.

The second context is Library of Congress's Chronicling America (LoCCA) collection, which contains over 19 million historical newspaper page scans. Library of Congress provides an OCR, but the quality is low (*49*). There is a large literature studying historical newspapers at scale, which overwhelmingly uses keyword search and does not unlock the power of large language models due to poor quality digitization (*20*). LoCCA elucidates how EffOCR: 1) performs in the highest resource setting, English; 2) extensibility across Latin and *kanji* characters, which differ significantly in their aspect ratios and complexity; 3) extensibility to the many Unicode renderable languages that use the Latin script.

Figure 2 shows example documents. Because seq2seq requires lines as inputs, lines are drawn randomly from the Japanese volumes and from 10 randomly selected newspapers in LoCCA. (EffOCR could be trained to localize characters in multiline inputs). Lines correspond to cells in tables and single lines within columns/rows in prose. The baseline training sets range from 291 lines for Chronicling America to 1309 cells for horizontal Japanese, highly feasible for researchers to label in an afternoon. The study's training datasets will be publicly released.

Measurement and Comparisons

OCR accuracy is measured using the character error rate (CER), the Levenshtein distance between the OCR'ed string and the ground truth, normalized by the length of the ground truth. A CER of 0.5, for instance, translates to mispredicting approximately half of characters.

The most widely used OCR engines are commercial products that do not support fine-tuning and have proprietary architectures. The study compares EffOCR to Google Cloud Vision (GCV) and Baidu OCR (popular for Asian languages). We also consider four open source architectures: EasyOCR's convolutional recurrent neural network (CRNN) framework (*48*), TrOCR's sequence-to-sequence encoder-decoder transformer (base and small) (*32*), Tesseract's bi-directional LSTM, and PaddleOCR's Single Vision Text Recognition (SVTR), which also abandons seq2seq, dividing text images into small (non-character) patches, using mixing blocks to perceive inter- and intra-character patterns, and recognizing text by linear prediction (*15*).

The pre-trained EasyOCR, PaddleOCR, and TrOCR models are fine-tuned on the same target data as EffOCR. Considerable resources have been devoted to pre-training these models (for example, TrOCR was pre-trained on 684 million English synthetic text lines), and hence comparisons elucidate performance when these pre-trained models are further tuned on the target datasets. For a more apples-to-apples comparison, the study examines the accuracy of these architectures when trained from scratch on 8,000 synthetic text lines (like EffOCR) and the same target crops. EasyOCR and PaddleOCR do not support vertical Japanese, and TrOCR does not support any Japanese. Tesseract offered little support for fine-tuning until recently and hence most of its applications have been off-the-shelf, which is this study's focus.

Results

EffOCR provides a highly accurate OCR with minimal training data, in contexts where current solutions fail. For vertical Japanese tables, the best EffOCR CER is 0.7% (Table 1). The next best alternative, Baidu OCR, has a CER of 55.6%, making nearly 80 times more errors. The best EffOCR CER is modestly higher for the Japanese prose (2.7%); these scans are low resolution and some characters are illegible, to provide a context where OCR with language modeling could offer a clear advantage. Yet EffOCR makes 5 times fewer errors than the next best alternative (GCV), whose CER of 13.5% will not support applications that require high accuracy. For horizontal Japanese - a higher resource setting - the EffOCR CER is 0.6%, whereas the next-best-alternative (Paddle OCR fine-tuned on target crops) makes more than five times more errors. The different EffOCR models produce strikingly similar results, despite the significant differences in architecture (convolutional versus transformer) and model size (9.3M to 112.5M parameters).

The CER (uncased) for the LoCCA newspapers is around 2%. GCV has the best performance (0.5%), followed by fine-tuned TrOCR (Base) (1.3% CER). The advantage of EffOCR on English - the quintessential high resource setting - is its open-source codebase and fast runtime. GCV makes significant layout errors when fed full newspaper page scans, which have complex layouts (*47*), and hence the performance in Table 1 cannot be replicated when it is fed scans. GCV charges per image, and the supplementary materials estimate a cost at current prices of $23 million USD to digitize LoCCA at the line image level, over several orders of magnitude more costly than deploying EffOCR-C (Small).

Table 1 examines CPU runtime for open source architectures, measured by lines processed per second on identical dedicated hardware (GPUs are prohibitively costly for mass digitization). EffOCR-C (Small) is 31 times faster than TrOCR (Base), which is likely to be cost prohibitive for large scale applications. EffOCR supports inference parallelization across characters - promoting faster inference - whereas seq2seq requires autoregressive decoding. On English, the most plausible scalable alternative is fine-tuned EasyOCR. With a third of the parameters of EffOCR-C (Small), inference is faster, but the CER is around 29% higher. For horizontal Japanese, EffOCR-C (Small) is three times more accurate and faster than PaddleOCR SVTR (fine-tuned), the next best alternative.

Figure 3 provides representative examples of errors, showing the target crop, the localized crop, and its five nearest neighbors, with the correct prediction highlighted in green. Errors tend to occur when the character is illegible or homoglyphic to another character (*e.g.* O and 0). For example, a 0 in one font can occasionally be indistinguishable from an O in another, an error that would be straightforward to correct in post-processing.

EffOCR's parsimonious vision-only architecture allows it to learn efficiently. This can be quantified with an apples-to-apples comparison, where EffOCR-C (Base) is compared to leading open source architectures pre-trained from scratch on the same number of synthetic text lines as EffOCR and tuned on the same target crops. Table 1 shows that EffOCR learns faster than both seq2seq (CRNN, TrOCR) and competing vision only (SVTR) architectures, with the next best alternative making 16 times more errors for Japanese and 4.7 times more errors for LoCCA. The transformer seq2seq model, which is extremely data hungry, is unusable.

Varying the number of target crops used for training further elucidates how efficiently the model learns. The supplementary materials document that on as few as 99 labeled table cells for Japanese and 21 rows for LoCCA (a 5% training data split) - creatable in a few minutes - EffOCR's CER is only 5% (Japanese) and 7% (English), showing viable few shot performance.

The supplementary materials report results from additional encoders, and examine how different ingredients of EffOCR contribute to its performance.

Using EffOCR to Liberate Data at Scale

EffOCR can convert the publications examined in this study (*25,26,51*) into a knowledge graph that provides rich information about network relationships in the historical Japanese economy.

These relationships can be observed through shareholding patterns, family ownership, financing, boards, occupational histories, family connections, spatial locations, and supply chains, combined with rich financial, production, and biographical information. Japanese individual and firm names are often only a few characters long, and hence a single OCR error can create a significant information bottleneck. Figure 4 provides an illustrative example of one component of this graph, showing supply chain networks in 1956 that were constructed by using EffOCR to digitize the customers and suppliers of Japan's 7,000 largest firms. Fine-grained control through EffOCR allowed detecting an atypical character separating firms in the customer-supplier lists - required for accurate digitization - that other OCR solutions did not systematically recognize, as well as accurate digitization of firm names. Each node in the graph is a firm, whose size is proportional to its degree centrality in the supply chain network. Shading denotes the big-three firms in pre-war Japan (Mitsui, Mitsubishi, and Sumitomo), as well as other firms - comprising Japan's largest conglomerates - targeted by the Holding Company Liquidation Commission in the late 1940s. The graph underscores that the largest pre-war firms remained the most central in Japanese supply chain networks in the 1950s, despite various policies in the late 1940s designed to curb their influence (*13, 14, 19, 41*). The many different types of relationships between individuals and firms accurately captured by EffOCR can facilitate a much more granular, comprehensive study of Japan's economic development than can be achieved with existing aggregated data.

Discussion

Indexing, analyzing, disseminating, and preserving diverse documentary history requires community engagement of stakeholders with the requisite fine-grained knowledge of the relevant settings. EffOCR facilitates this engagement because it is highly extensible to low-resource settings, sample-efficient to customize, and simple and cheap to train and deploy. In contrast, seq2seq is more aligned with the commercial objective of designing a product that is difficult for competitors to imitate. For example, EffOCR can be trained in the cloud with free student compute credits, whereas TrOCR required training on a multi-million dollar cluster with 32 32GB V100 cards. Lower resource languages may lack the pre-trained large language models required to initialize a transformer seq2seq model like TrOCR, and compute resources and data for training are also likely unavailable.

EffOCR encourages community engagement by combining the parsimonious conceptualization of OCR from nearly a century ago with the AI revolution, integrating the follow features:

Character level: EffOCR creates semantically rich visual embeddings of individual characters, a parsimonious problem. Annotators can select which of the most probable character predictions from the pre-trained recognizer are correct, potentially using a simple mobile interface, or line level labels can be mapped to the character level once a localizer has been developed.

Language Extensibility: Language modeling advances have concentrated around less than two dozen modern languages, out of many thousands (*29*). Omitting the language model makes

EffOCR extensible and easy-to-train. To extend EffOCR to a new language, all one needs are renders for the appropriate character set. Additionally, characters do not need to be seen in sequence during training, so new characters can be added at inference time, valuable for archaeological contexts where new characters are regularly discovered. Omitting the language model makes it easy to mix scripts, necessary for some languages. The recognizer can also be exposed to characters in training using any desired sequencing. This is not true of multilingual seq2seq training, which leads to many OCR errors with endangered languages (*44*).

Decoupling localization and recognition: Theoretically, localization and recognition (akin to classification) may rely on different features of the image, suggesting modularity (*50*). Practically, decoupling allows localization and recognition to use different training sets, economizing on annotation costs since these tasks can require very different numbers of labels depending on the script. It also encourages community innovation and future-proofness, because it simplifies training recipes and makes it straightforward to swap encoders as the literature advances.

Open source: EffOCR will be integrated into Layout Parser (*47*), an open-source python package providing tools for off-the-shelf and customized document image analysis that is widely used in the social science community.

Scalable: EffOCR-C (Small) achieves fast CPU inference that can scale cheaply to hundreds of millions of documents.

Limitations and Future Directions: EffOCR is not currently an off-the-shelf solution. Rather, it is designed for contexts where researchers need fine-grained control. More extensive pre-training would be necessary to evaluate EffOCR's zero-shot capabilities. Its sample efficiency suggests the feasibility of designing an EffOCR that works well for many applications off-the-shelf by exposing the pre-trained model to a modest number of crowd-sourced crops from a wide range of documents. Another next step is to build upon the data augmentation and style transfer literatures (*2, 52*) to generate more diverse synthetic pre-training data.

This study does not focus on handwriting due to space constraints, but the approach would be analogous. Synthetic handwriting generators, *e.g.* (*4*), could provide extensive data for pre-training, analogous to this study's use of digital fonts.

There are some settings where EffOCR's framework is not suitable. If large portions of a document are illegible, language can provide educated guesses. Moreover, the heavy use of ligatures in some character sets and handwriting could lead to more challenging character localization. However, modern AI object detection methods are powerful and synthetic pre-training can encourage robustness to localization noise.

An emerging literature explores foregoing OCR altogether to directly reason on images (*18, 38, 43, 45*). The main focus is on commercial applications like receipts and train tickets. There are not end-to-end models for performing most research tasks, and document images are large to store. Hence, OCR is likely to remain relevant for academic applications and digital archives for the foreseeable future. EffOCR's simple character image retrieval framework can expand accessibility to a rich diversity of human knowledge.

Acknowledgments

Funding: Harvard Data Science Initiative. National Science Foundation Award 1823616

Competing Interests: We have no competing interests to disclose.

Data and Materials Availability: Code for this study is available at (*7*) and (*8*).

List of Supplementary Materials

Materials and Methods
Supplementary Text
Tables S1 and S2
Figure S1
References

Tables

					Character Error Rate				Lines/second	
Model/Engine	Seq2Seq?	Transformer?	Pretraining	Parameters	Horiz. Jap.	Vertical Jap. (tables)	Vertical Jap. (prose)	Chron. America	Horiz. Jap.	Chron. America
EffOCR-C (Base)	×	×	from scratch	112.5 M	**0.006**	**0.007**	0.030	0.023	0.79	0.49
EffOCR-C (Small)	×	×	from scratch	9.3 M	0.010	0.009	0.036	0.028	19.46	13.40
EffOCR-T (Base)	×	✓	from scratch	101.8 M	0.009	0.007	**0.027**	0.022	0.19	0.31
Google Cloud Vision OCR	?	?	off-the-shelf	?	0.173	0.695	0.135	**0.005**	?	?
Baidu OCR	?	?	off-the-shelf	?	0.060	0.556	0.177	-	?	?
Tesseract OCR (Best)	✓	×	off-the-shelf	1.4 M	1.021	0.996	0.744	0.106	4.90	4.47
EasyOCR CRNN	✓	×	off-the-shelf	3.8 M	0.191	-	-	0.170	33.55	19.80
			fine-tuned		0.082	-	-	0.036		
			from scratch		0.132	-	-	0.131		
PaddleOCR SVTR	×	×	off-the-shelf	11 M	0.085	-	-	0.304	13.34	13.56
			fine-tuned		0.032	-	-	0.103		
			from scratch		0.097	-	-	0.104		
TrOCR (Base)	✓	✓	off-the-shelf	334 M	-	-	-	0.015	-	0.43
			fine-tuned		-	-	-	0.013		
			from scratch		-	-	-	0.809		
TrOCR (Small)	✓	✓	off-the-shelf	62 M	-	-	-	0.039	-	0.97
			fine-tuned		-	-	-	0.075		
			from scratch		-	-	-	0.773		

Table 1: **Baseline Results and Comparisons.** This table reports the performance of different OCR architectures, *off-the-shelf* (without fine-tuning on target data), *fine-tuned* on the target publication training set from a pre-trained OCR checkpoint, and trained *from scratch* on synthetic text lines and the target publication training set. “?” indicates that the field is unknown due to the proprietary nature of the architecture.

Figures

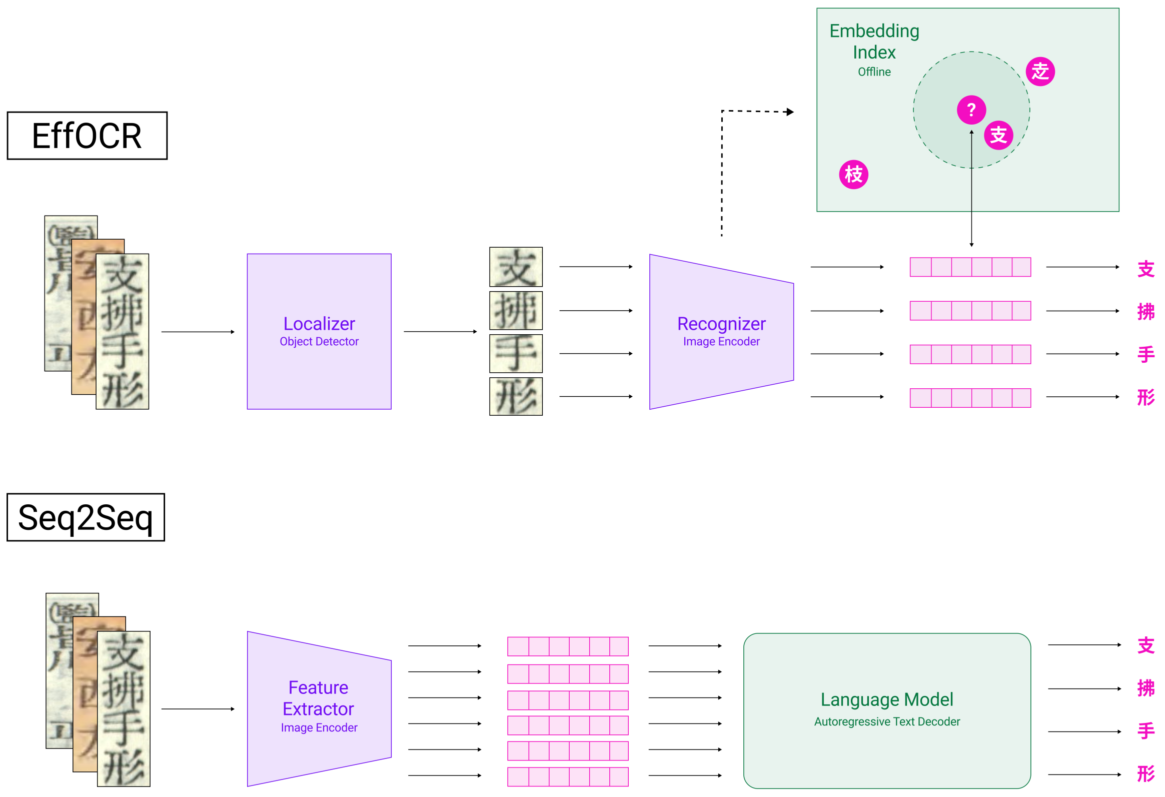
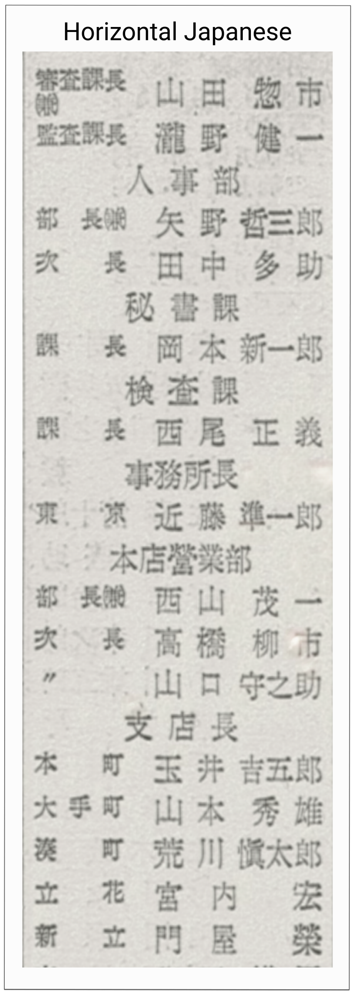
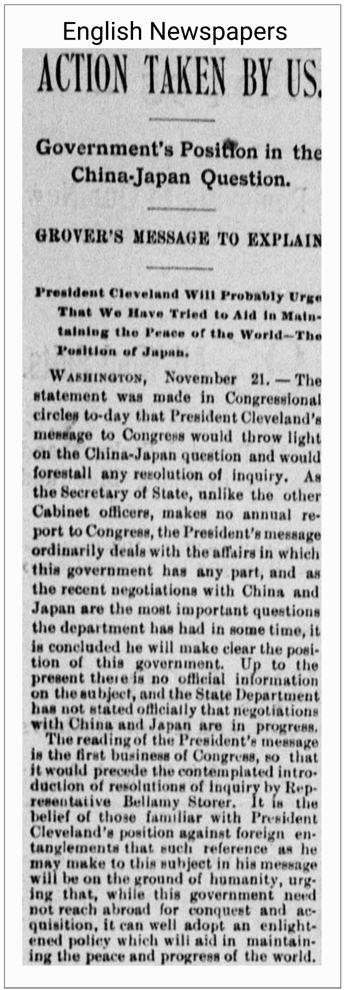


Figure 1: **EffOCR and Seq2Seq Model Architectures.** This figure represents the EffOCR architecture, as compared to a typical sequence-to-sequence OCR architecture.


Horizontal Japanese

審査課長(兼) 山田 惣市
監査課長 瀧野 健一
人事部
部長(兼) 矢野 哲三郎
次長 田中 多助
秘書課
課長 岡本 新一郎
検査課
課長 西尾 正義
事務所長
東京 近藤 準一郎
本店營業部
部長(兼) 西山 茂一
次長 高橋 柳市
〃 山口 守之助
支店長
本町 玉井 吉五郎
大手町 山本 秀雄
湊町 荒川 愼太郎
立花 宮内 宏
新立 門屋 榮


English Newspapers

ACTION TAKEN BY US.

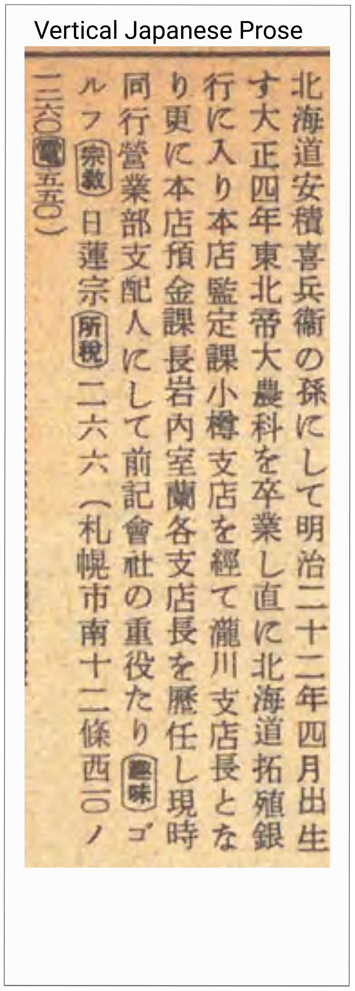
Government's Position in the China-Japan Question.

GROVER'S MESSAGE TO EXPLAIN

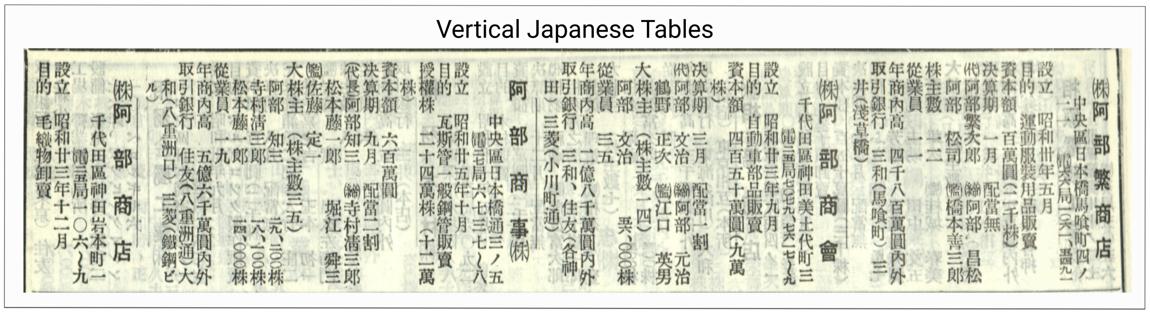
President Cleveland Will Probably Urge That We Have Tried to Aid in Maintaining the Peace of the World—The Position of Japan.

WASHINGTON, November 21.—The statement was made in Congressional circles to-day that President Cleveland's message to Congress would throw light on the China-Japan question and would forestall any resolution of inquiry. As the Secretary of State, unlike the other Cabinet officers, makes no annual report to Congress, the President's message ordinarily deals with the affairs in which this government has any part, and as the recent negotiations with China and Japan are the most important questions the department has had in some time, it is concluded he will make clear the position of this government. Up to the present there is no official information on the subject, and the State Department has not stated officially that negotiations with China and Japan are in progress.

The reading of the President's message is the first business of Congress, so that it would precede the contemplated introduction of resolutions of inquiry by Representative Bellamy Storer. It is the belief of those familiar with President Cleveland's position against foreign entanglements that such reference as he may make to this subject in his message will be on the ground of humanity, urging that, while this government need not reach abroad for conquest and acquisition, it can well adopt an enlightened policy which will aid in maintaining the peace and progress of the world.


Vertical Japanese Prose

北海道安積喜兵衛の孫にして明治二十二年四月出生す大正四年東北帝大農科を卒業し直に北海道拓殖銀行に入り本店監定課小樽支店を經て瀧川支店長となり更に本店預金課長岩内室蘭各支店長を歴任し現時同行營業部支配人にして前記會社の重役たり㊇ゴルフ㊡日蓮宗㊥二六六（札幌市南十二條西一〇ノ一三六〇㊢五五〇）


Vertical Japanese Tables

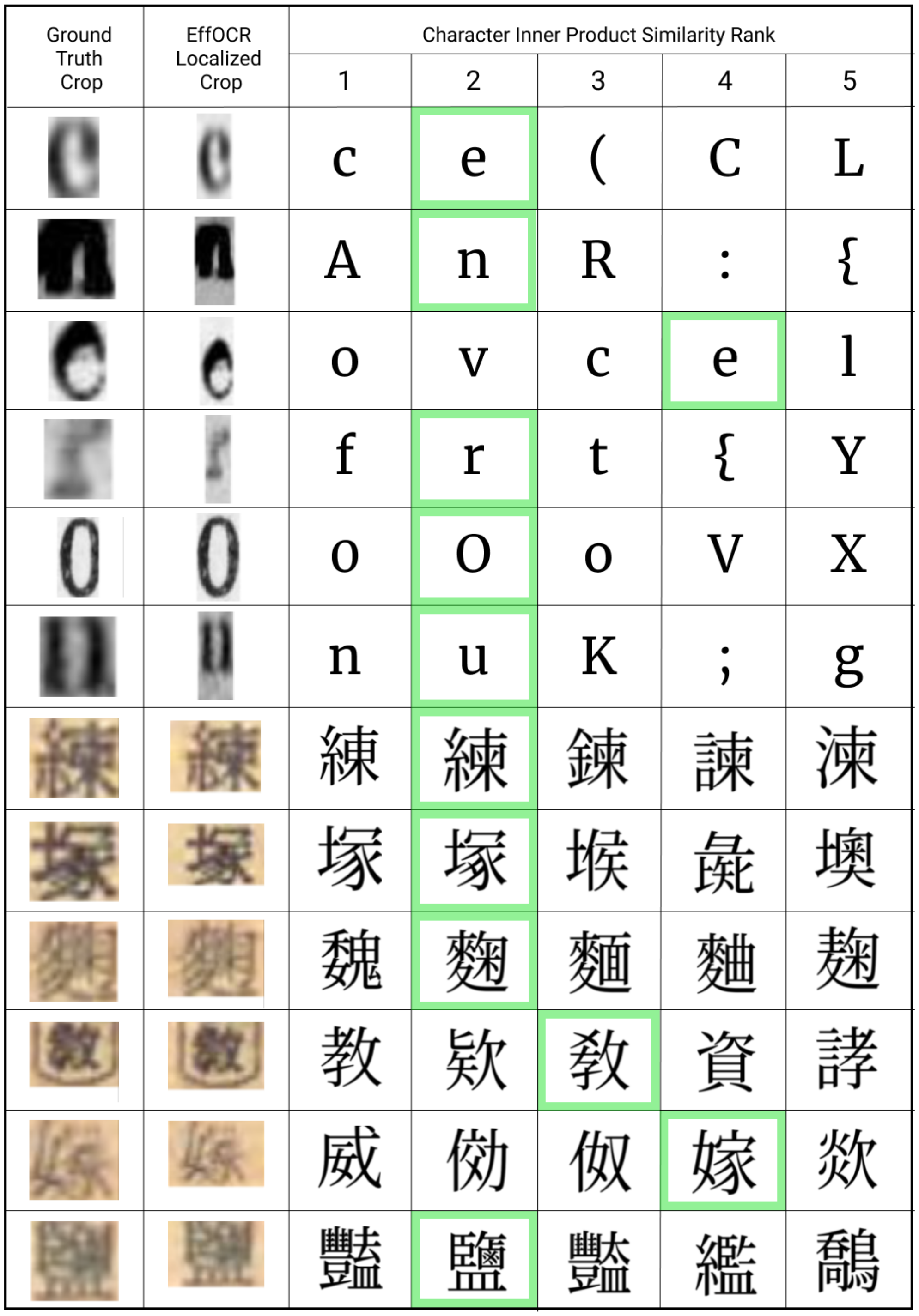
(株)阿部繁商店
中央區日本橋馬喰町四ノ二一 (電)茅(66)局一〇六二、一四九一
設立　昭和卅年五月
目的　運動服裝用品販賣
資本額　百萬圓（二千株）
決算期　一月　配當無
(代)阿部繁次郎　(締)阿部昌松
大阿部松司　(監)橋本善三郎
株主數　一二
從業員　一一
年商内高　四千八百萬圓内外
取引銀行　三和（馬喰町）三井（淺草橋）

(株)阿部商會
千代田區神田美土代町三 (電)(25)局七〇七九、七六一七～九
設立　昭和廿三年九月
目的　自動車部品販賣
資本額　四百五十萬圓（九萬株）
決算期　三月　配當一割
(代)阿部文治　(締)阿部元治
鶴野正次　(監)江口英男
大株主（株主數一四）
阿部文治　三六、〇〇〇株
從業員　三五
年商内高　二億八千萬圓内外
取引銀行　三和、住友（各神田）三菱（小川町通）

阿部商事(株)
中央區日本橋通三ノ五 (電)(27)局六七三七～八
設立　昭和廿五年十月
目的　瓦斯管一般鋼管販賣
授權株　二十四萬株（十二萬株）
資本額　六百萬圓
決算期　九月　配當二割
(代長)阿部知三　(締)寺村清三郎
松本藤一郎　堀江舜三
(監)佐藤定一
大株主（株主數三五）
阿部知三　二九、三〇〇株
寺村清三郎　一八、一〇〇株
松本藤一郎　一四、〇〇〇株
從業員　一九
年商内高　五億六千萬圓内外
取引銀行　住友（八重洲通）大和（八重洲口）三菱（鐵鋼ビル）

(株)阿部商店
千代田區神田岩本町一 (電)(25)局一〇六～九
設立　昭和廿三年十二月
目的　毛織物卸賣

Figure 2: **Dataset Description.** Representative samples of the publications examined in this study.



Ground Truth Crop	EffOCR Localized Crop	Character Inner Product Similarity Rank				
		1	2	3	4	5
		c	🟩 e	(	C	L
		A	🟩 n	R	:	{
		o	v	c	🟩 e	l
		f	🟩 r	t	{	Y
		o	🟩 O	o	V	X
		n	🟩 u	K	;	g
		練	🟩 練	鍊	諫	湅
		塚	🟩 塚	堠	彘	墺
		魏	🟩 麴	麵	麯	趜
		教	欸	🟩 敎	資	誟
		威	傚	假	🟩 嫁	欻
		豔	🟩 鹽	豔	繿	鷊

Figure 3: **Error Analysis.** Representative examples of EffOCR errors, showing the target crop, the EffOCR localized crop, and the five nearest characters in the embedding index, with the correct character highlighted in green.

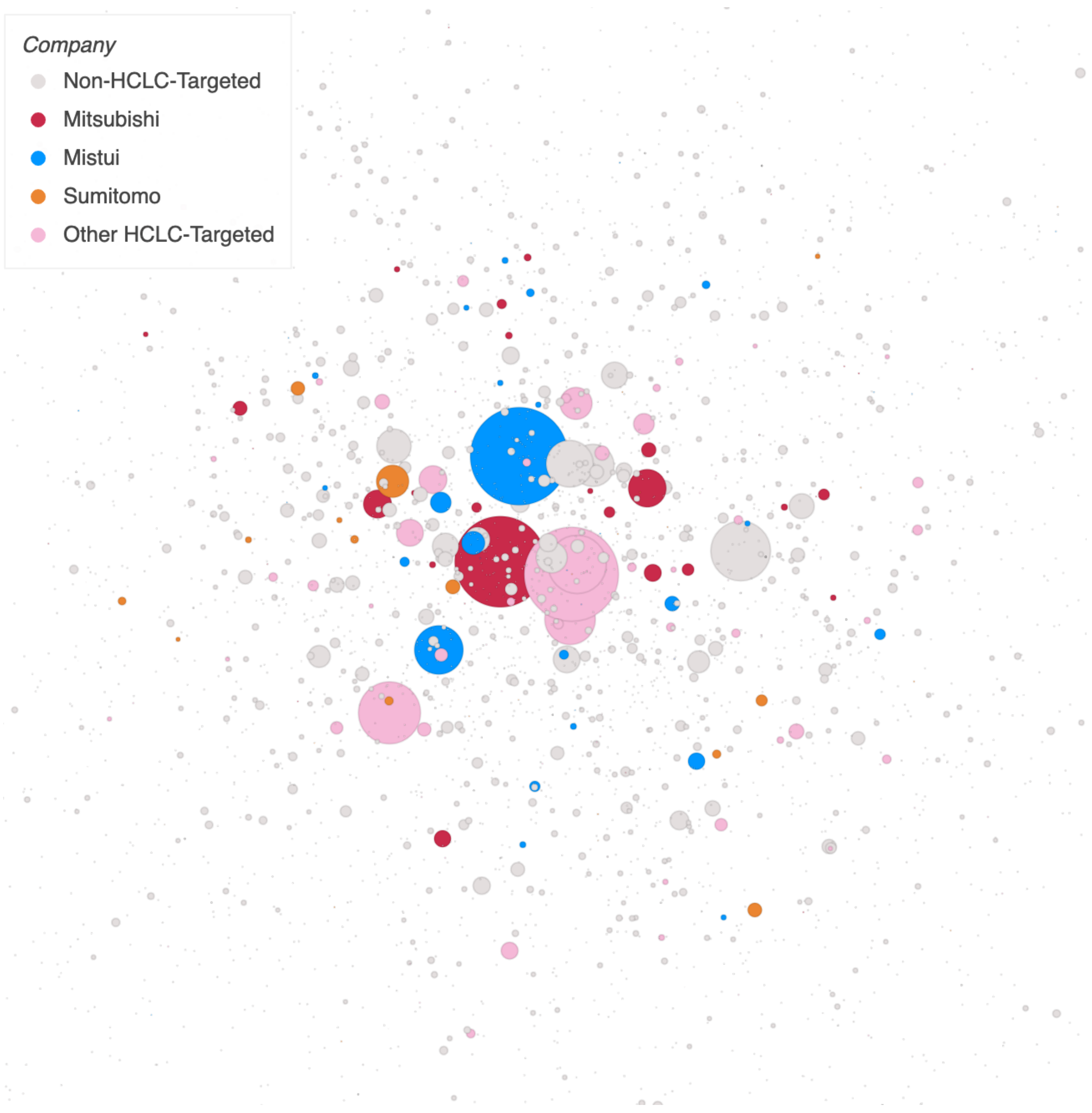


Figure 4: **Supply Chain Networks (Japan, 1956).** Each node in the graph is a firm, whose size is proportional to its degree centrality in the supply chain network. Shading denotes three of the largest firms in pre-war Japan - Mitsui, Mitsubishi, and Sumitomo - as well as other firms - comprising Japan's largest conglomerates - targeted by the Holding Company Liquidation Commission (HCLC) in the late 1940s.

Supplementary materials

Materials and Methods

Overview

EffOCR's architecture draws inspiration from metric learning methods for efficient image retrieval (*16*), joining a recent literature on self-supervision through simple data augmentation for image encoders (*10, 11, 17*). EffOCR contrastively trains the encoder - using a Supervised Contrastive loss function (*30*) - to learn 768-dimensional dense vector representations of character crops. Crops of the same character - regardless of style - are projected nearby, and crops of different characters are projected further apart. Embedded characters are decoded by retrieving their nearest neighbor from an offline index of exemplar character image embeddings. Distances are computed using cosine similarity with a Facebook Artificial Intelligence Similarly Search (FAISS) backend (*28*). The vision embeddings alone are sufficient to infer text since they represent characters - not text lines like in seq2seq - and hence decoding them does not require a language model or learned parameters.

The closest frameworks to EffOCR in their overall design are the original OCR conceptualizations, such as Tauschek's 1920s reading machine, which used human engineered features rather than neural networks to recognize localized characters. More recently, CharNet (*57*), developed for scene text (not documents), uses separate convolutional networks for dense classification and regression at a single scale, outputting a character class and bounding box at every spatial location, and then aggregates this information with confidence scores to make final predictions. EffOCR in contrast deploys widely used, highly optimized object detection methods to localize characters and then feeds character crops to a contrastively trained recognizer.[1]

Encoders

Different encoders can be used interchangeably for EffOCR's character localization module (hereafter, "localizer") and character recognizing module (hereafter "recognizer"). We use the following:

- **EffOCR-C (Base)**: ConvNeXt (Tiny) (*36*) for both the localizer and recognizer. Both models are initialized from the officially released checkpoint with specifications:
  {`size: "tiny"`}
- **EffOCR-T (Base)**: XCiT (Small) (*3*) for both the localizer and recognizer. Both models are initialized from the officially released checkpoint with specifications:
  {`size: "small", depth: 12, patch_size: 8, resoultion: 224`}
- **EffOCR-C (Small)**: YOLOv5 (Small) (*27*) for the localizer and MobileNetV3 (Small) (*23*) for the recognizer. YOLOv5 is initialized from the officially released `YOLOv5s`

[1]Others have also used contrastive learning for OCR, in particular (*1*) use a self-supervised, sequence-to-sequence contrastive learning approach.

checkpoint, and MobileNetV3 is initially from the PyTorch Image Models ("timm") (*54*) produced checkpoint with specifications:
`{size: "small", channel_multiplier: 0.50}`

For ablations, we also examine:

- Swin (Tiny) (*35*) for both the localizer and recognizer. Both models are initialized from the officially released checkpoint with specifications:
  `{size: "tiny", patch_size: 4, window: 7, resolution: 224}`
- ViTDet (Base) (*33*) for the localizer and a vanilla vision transformer, ViT (Base), for the recognizer. Both models are initialized from the officially released checkpoint with specifications:
  `{size: "base", patch_size: 16, resolution: 224}`

These architectures were selected for the following reasons:

- **EffOCR-C (Base)**: ConvNeXt is a new state-of-the-art CNN backbone, in contrast to the other three vision transformer encoders.
- **EffOCR-T (Base)**: XCiT was chosen because of its comparative advantage in modeling fine-grained features via the ability to accommodate smaller patch sizes through a linear complexity attention mechanism, which may be especially suitable for character images with small spatial extents (as measured in pixels).
- **EffOCR-C (Small)**: MobileNetV3 (Small) and YOLOv5 (Small) were collectively chosen to produce a speed optimized EffOCR, as both architectures are popular, easily customizable, and speed-optimized by design.
- The Swin transformer was selected because of its state-of-the-art performance on object detection tasks.
- The original ViT embeddings perform well for image retrieval, and have become a new baseline for image retrieval (*16*).

The inference speed advantages offered by a smaller transformer encoder, such as MobileViT, are much more modest than that offered by MobileNetV3, and hence an EffOCR-T (small) model is not developed, although it would be straightforward to do so should users desire it. In tests, a MobileViTv2 (small) Recognizer model was approximately 6.5 times slower than a comparable MobileNetv3 Recgonizer.

As the deep learning literature advances and new models are developed, EffOCR's modular framework and simple training recipes make it straightforward to swap in new encoders, granting the model a degree of future-proofness.

These models are all trained on a single A6000 GPU card, with hyperparameters selected using the 15% validation split, save for the models with XCiT (Small) or ViT (Base) encoders, which were trained on two A6000 GPU cards.

Character Localization

All models use an MMDetection (*9*) backend for localization, except for the ViTDet ablation, which uses Detectron2 (*56*) and YOLOv5 (Small) (*27*) for EffOCR-C (Small), which uses its own custom training scripts. Only one EffOCR configuration, EffOCR-C (Small), has a localizer that uses a one-stage object detection framework: YOLOv5 (Small) (*27*). All others use a two-stage object detector, specifically a Cascade R-CNN (*6*). One stage object detection is faster, and hence makes sense for the small model, where a central objective is fast inference speed.

The localizers built with ConvNeXt (EffOCR-C Base), XCiT (EffOCR-T Base), and Swin (ablation) are trained on 8,000 textlines of synthetic data for 40 epochs at a constant learning rate of $1e-4$ and fine-tuned on benchmark data for 100 epochs at a $2.5e-5$ constant learning rate, all with anchor generator scales $[2, 8, 32]$. ViTDet is trained on 8,000 textlines of synthetic data for 40 epochs with a constant learning rate of $1e-4$, and then fine-tuned for 100 epochs on benchmark data with a $1e-5$ constant learning rate. The YOLO localizer is trained on 8,000 textlines of synthetic data for 30 epochs at a constant learning rate of $1e-2$ and fine-tuned on benchmark data for 30 additional epochs, still at a constant $1e-2$ learning rate.

The synthetic data used for pre-training the localizers and comparison models was created using a custom synthetic data generator, which can found at the "EffSynth" GitHub repository (*8*). This generator was used to create six synthetic dataset variants, each consisting of 10,000 synthetic lines with an 80%-10%-10% train-test-validation split. The six dataset variants are: horizontal English with character sequences generated at random, horizontal Japanese with character sequences generated at random, vertical Japanese with character sequences generated at random, horizontal English with text sequences generated from Wikipedia, horizontal Japanese with text sequences generated from (Japanese) Wikipedia, and vertical Japanese with text sequences generated from (Japanese) Wikipedia. Text sequence based synthetic datasets were used to pre-train seq2seq models that rely on language context, e.g., TrOCR and CRNN; character sequence based synthetic datasets were used to pre-train non-seq2seq models, e.g., EffOCR and SVTR.

Character Recognition

The EffOCR recognizer is trained using the Supervised Contrastive ("SupCon") loss function (*30*), a generalization of the InfoNCE loss (*42*) that allows for multiple positive and negative pairs for a given anchor. In particular, we work with the "outside" SupCon loss formulation

$$\mathcal{L}^{\text{sup}}_{\text{out}} = \sum_{i\in I} \mathcal{L}^{\text{sup}}_{\text{out},i} = \sum_{i\in I} \frac{-1}{|P(i)|} \sum_{p\in P(i)} \log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_p/\tau)}{\sum_{a\in A(i)} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_a/\tau)}$$

as implemented in PyTorch Metric Learning (*39*), where τ is a temperature parameter, i indexes a sample in a "multiviewed" batch (in this case multiple fonts/augmentations of characters with the same identity), $j(i)$ is the index of a positive sample (in this case other characters that share the same identity), $P(i)$ is the set of indices of all positives in the multiviewed batch that are distinct from i, $A(i)$ is the set of all indices excluding i, and z is an embedding of a sample in the batch (*30*).

To create training batches for the recognizer, EffOCR uses a custom m per class sampling algorithm *without replacement* adapted from the PyTorch Metric Learning repository (*39*). This metric learning batch sampling algorithm also implements batching and training with hard negatives, where the negative samples in a batch are selected to be semantically close to one another, and thus contrasts made between anchors and hard negatives may be especially informative for the model to update on. Indeed, one of the main advantages of contrastive training is that it allows the learning process to exploit hard negative mining.

More specifically, the custom batch sampling algorithm samples m character variants for each class (character) - drawn from both target documents and augmented digital fonts. We choose $m = 4$ and the batch size is 128, meaning 4 styles/representations of each of 32 different characters appear in each batch. The model learns to map character crops of the same identity to similar dense vectors in a semantically rich, high-dimensional vector space, and vice versa. There is no natural definition of an epoch in the context of batch-based sampling for contrastive learning with data augmentation in the way EffOCR formulates this procedure. For EffOCR recognizer training, an epoch is defined as some number P passes through all unique characters N in the character set under consideration, i.e., $N = 13,738$ for Japanese and $N = 91$ for English. Empirically, a good setting for Japanese is $P = 1$, so the total number of classes in an epoch is 13,738, and for English $P = 10$, so the total number of classes in an epoch is 910. Sampling for each class occurs without replacement, for better coverage of character variants. Because of this, the number of passes P matters, as it determines the number of character variants used for contrastive training in each epoch.

Every character crop that appears in the training set is embedded using a model first trained without hard negative mining/sampling, and for each we find its 8 nearest neighbors. The EffOCR recognizer is then trained again from scratch, with batches being sampled with an m per class sampler (without replacement) that is further modified to randomly intersperse hard negative sets (8 nearest neighbor characters, $m = 4$ variants of each) throughout batches.

EffOCR is trained on digital font renders from readily available fonts (13 for Japanese and 14 for English), along with a modest number of labeled crops from the target datasets.[2] The digital fonts are augmented by randomly applying affine transformations (translation and scaling); background coloring, color jittering, color inversion, and grayscaling; and Gaussian blurring. The model trains on digital fonts and labeled crops *together*, since the objective is to learn general purpose embeddings that would map target crops nearby to digital renders. All recognizer models except MobileNetV3 use an AdamW optimizer with weight decay of $5e-4$, a SupCon loss with temperature of 0.1, a learning rate of $2e-5$, and a batch size of 128. MobileNetv3 uses the same parameters except a learning rate of $2e-3$. The Japanese datasets are trained for 60 epochs and the English dataset for 30.

[2]Fonts for Japanese included: Dela Gothic One Regular; Hachi Maru Pop Regular; Hina Mincho Regular; Komorebi Gothic; Kosugi Regular; New Tegomin Regular; Noto Serif CJK JP Regular; Reggae One Regular; Shippori Mincho B1 Regular; Stick Regular; taisyokatujippoi7T5; Tanugo Regular; and Yomogi Regular. Fonts for English included: Anton Regular; Cutive Mono Regular; EB Garamond Regular; Fredoka Regular; IM Fell DW Pica Regular; NewYorker-jLv; Noto Serif Regular; Oldnewspapertypes-449D; Orbitron Regular; Special Elite Regular; Ultra Regular; VT323 Regular; ZaiConsulPolishTypewriter-MVAxw; and ZaiCourierPolski1941-Yza4q. See the EffOCR GitHub repository for the font files themselves (*7*).

After recognizer training is completed, the recognizer is used as an encoder to create an offline index of exemplar character embeddings to be searched at inference time for the purposes of character recognition. Specifically, the exemplar character embedding index is created by embedding image renders for all the unicode characters supported by the Google Noto Serif font series, i.e., Noto Serif CJK JP Regular for models trained for Japanese OCR and Noto Serif Regular for models trained for English OCR. The Google Noto series is chosen as an exemplar font due to both its extremely wide coverage of glyphs and the simplicity of its style, though, by virtue of EffOCR's training, other fonts could be used as well. At inference time, FAISS (*28*) is used to perform an *inner product* similarity search that compares character embeddings in the sample being inferenced to exemplar character embeddings in this offline index; identities are assigned to inferenced characters using the identity of that character's nearest neighbor in the offline exemplar index, i.e., k-NN classification with $k = 1$.

For case sensitive applications, EffOCR character recognition for English text can also be lightly post-processed to help better differentiate uppercase and lowercase letters from one another: one can force a character to be uppercased or lowercased through simple rules based statistics about the dimensions of bounding boxes (in the sample undergoing inference). This procedure is irrelevant for results reported in this text, however, for which CER is measured uncased.

Checkpoints/weights for all recognizers are supported by implementations from timm (*54*).

Comparisons

To examine sample efficiency, we train alternative architectures from scratch, on the same number of synthetic text lines used to train EffOCR. Specifically, the comparison architectures are, as applicable, initialized with "default" pre-trained checkpoints that have not yet been exposed to an OCR task, e.g., masked language model pre-trained weights for text transformers or ImageNet pre-trained weights for CNNs and vision transformers. These comparison architectures are then trained on 8,000 synthetic text lines per the applicable synthetic dataset variant (see: Methods - Synthetic Data) as a form of standardized OCR-task-specific pre-training. They are then fine-tuned on the same benchmark datasets used to assess EffOCR, but with varying train-test-validation splits: 70%-15%-15%, 50%-25%-25%, 20%-40%-40%, 5%-47.5%-47.5%, and 0%-50%-50% (i.e., zero-shot).

The hyperparameters used for initializing and training comparison models are as follows:

- The EasyOCR implemented **CRNN** (*48*) comparison is trained from a random initialization (as is the default in EasyOCR) for 100,000 iterations on the horizontal English text sequence and horizontal Japanese text sequence synthetic datasets, respectively. The learning rate is fixed at 1.0 with an Adadelta optimizer and the batch size is 128, per the EasyOCR configuration defaults. The architecture uses VGG for feature extraction, a BiLSTM for seq2seq/language modeling, and a CTC loss, as also is the EasyOCR default. A new prediction head is used to match the character set associated with EffOCR for Japanese. The resulting model is then fine-tuned for 30,000 iterations with a batch size of 64, and all other hyperparameters the same, on the benchmark datasets of varying splits.

- The **SVTR** (*15*) comparison is first trained from a random initialization for 500 epochs with an Adam optimizer with cosine-scheduled learning rate of 0.001 and batch size of 32 on horizontal English character sequence and horizontal Japanese character sequence synthetic datasets, respectively. All these hyperparameters are PaddleOCR defaults, which are also used for fine-tuning on the benchmark dataset splits.

- The **TrOCR** (*32*) comparison models are initialized from the appropriate vision transformer and language transformer pre-trained encoder and decoder checkpoints: for TrOCR (Base) this is the officially released BEiT (Base) checkpoint and the officially released RoBERTa (Large) checkpoint used by the TrOCR authors for model initialization; for TrOCR (Small) these are similarly the officially released checkpoints for DeiT (Small) and MiniLM used by the TrOCR authors for their model initialization. These checkpoints are exported directly from the TrOCR GitHub repository (*31*) using a modified script originally authored by Hugging Face (*55*), such that training is possible in native PyTorch with Huggingface model implementations. TrOCR (Base) is trained on the horizontal English synthetic text sequence dataset for 60 epochs at a fixed learning rate of $5e - 7$ with a batch size of 16; TrOCR (Small) is trained for 40 epochs, with all other hyperparameters the same. (The learning rate was selected based on experiments with the validation set.) The resulting models are then fine-tuned with the same hyperparameters on the various benchmark dataset splits.

To evaluate how existing solutions perform when fine-tuned on the EffOCR benchmark datasets, existing pre-trained checkpoints from the EasyOCR CRNN, PaddleOCR SVTR, and TrOCR (Base) and TrOCR (Small) models are fine-tuned on the baseline 70%-15%-15% split of the benchmark datasets. Specifically, the 15% validation set is used for hyperparameter tuning and the 15% test set is used to construct the results reported in the study.

For all comparison models, training hyperparamters are the same as used during the sample efficiency assessments with standardized synthetic pre-training, save that prediction heads for relevant models are left as they are by default. Model initialization differs, accordingly: TrOCR (Base) and TrOCR (Small) use `microsoft/trocr-base-stage1` and `microsoft/trocr-small-stage1` checkpoints, respectively; EasyOCR CRNN uses the most recently released `japanese_g2.pth` and `english_g2.pth` checkpoints; and PaddleOCR SVTR uses the most recently released `japan_PP-OCRv3_rec_train` and `en_PP-OCRv3_rec_train` best accuracy checkpoints.

Inference Speed Comparisons

For digitizing large-scale collections, fast inference on a CPU is necessary, due to the high costs of GPU compute. All comparisons are made on four 2200 MHz CPU cores, selected to represent a plausible and relatively affordable research compute setup. To standardize measurements of speed, each model generated predictions on the same 15% test set. All EffOCR models are implemented with ONNX Runtime for cross-compatibility and speed.

Inference speed is inherently dependent on implementation and it is plausible that the other open-source architectures may be updated in the future to achieve faster inference speeds. A strong correlation between model size and inference speed is apparent and intuitive, highlighting the utility of the EffOCR-C (Small) model for digitizing knowledge - like the Chronicling America collection - at scale.

A random sample of 10 LoCCA scans shows an average of 1944 column x lines per scan (historical newspapers used small fonts and contained few images), which implies the cost at current prices to digitize the LoCCA collection at the line level using GCV would be over 23 million US dollars.

Using FS4 VM instances in Microsoft Azure to process all content in the LoCCA collection for one randomly selected day per decade, on average it took 17.21 seconds to process 1,000 lines with EffOCR-C (small). At current prices, this translates to a cost of \$0.000908 per one thousand lines, as compared to GCV's current prices of \$1.50 (first 5 million units) and \$0.60 (above 5 million units) per thousand lines to process Chronicling America at the line level.

Benchmark Dataset Creation

The OCR systems evaluated in this study take lines (cells in tables or individual lines from columns in prose) as inputs. These segments were created using a Mask R-CNN (*21*) model custom-trained with Layout Parser (*47*), an open-source package that provides a unified, deep learning powered toolkit for recognizing document layouts. Mask R-CNN was applied to the three Japanese publications considered and to ten different newspapers randomly selected from Chronicling America. Segments were selected at random for inclusion in this study's benchmark datasets. Table S-1 provides dataset statistics.

To create the character region and text annotations, three highly skilled annotators - undergraduate and graduate students - annotated each segment. All discrepancies were then hand checked and resolved by the study authors. Each of the datasets has a 70%-15%-15% train-validate-test split used for baseline evaluations. The validation set was used for model development, whereas the test set was used only once, to create the results reported in this study.

Supplementary Text

Ablations

To elucidate which components of EffOCR are essential for its performance, several ablations are examined in Table S-2: using a simple feedforward neural network classifier head for recognition instead of performing k-nearest neighbors classification[3], training with and without hard negatives, disabling training on synthetic data for the recognizer and localizer, and the use of alternative vision encoders. All ablations use a fixed set of hyperparameters that are associated with a specific localizer-recognizer configuration; these hyperparameters are outlined in the sections on Character Localization and Character Recognition.

[3]Implicitly, retrieving the nearest neighbor character from an index of offline exemplar character embeddings, as the EffOCR recognizer does by default, is k-NN classification with $k = 1$.

Modeling character-level classification as an image retrieval problem weakly dominates the classification performance when using a standard multilayer perceptron with softmax procedure for classification. OCR as retrieval is chosen as the baseline not only due to its performance, but because it also allows for adding new characters at inference time (just embed a new exemplar character and add it to the offline index) - common in historical and archaeological settings - and because efficient similarity search technologies like FAISS (*28*) provide fast inference.

Removing hard negatives increases the character error rate substantially, particularly for Japanese, which has many characters with highly similar visual appearances, e.g., some multi-stroke kanji are nearly identical to one another and differ only in the slants of some strokes. Using hard negatives in constrastive training effectively incentivizes the model to distinguish between these very visually similar characters.

Training on only labels from the target documents leads to a large deterioration in performance for Japanese. This is as expected, given that only a fraction of *kanji* characters appear in the small training datasets. The deterioration in performance is modest for English, where there are far fewer characters. The opposite is true for character localization. Localization for English is a harder problem than for Japanese because character silhouettes and aspect ratios are more variable.

Two additional vision transformer encoders are explored: Swin (Tiny) (*35*) for both the localizer and recognizer and ViTDet (Base) (*33*) for the localizer and a vanilla vision transformer, ViT (Base), for the recognizer. The performance is similar to the base EffOCR-C and EffOCR-T models.

Sample Efficiency

To examine how efficiently EffOCR learns in comparison to leading open source architectures, we train different OCR models from scratch using varying amounts of annotated data. EffOCR-C (Base) is compared to SVTR (implemented via PaddleOCR) (*15*), CRNN (implemented via EasyOCR) (*48*), and TrOCR (*32*). All architectures are pre-trained from scratch on 8,000 synthetic text lines, starting from pre-trained checkpoints not customized for OCR when supported by the framework. They are then fine-tuned on the study's benchmark datasets, with varying train-test-validation splits: 70%-15%-15%, 50%-25%-25%, 20%-40%-40%, 5%-47.5%-47.5%, and 0%-50%-50% (i.e., zero-shot). These exercises are performed for LoCCA and horizontal Japanese, as vertical Japanese is not supported by the comparison architectures.

Figure S-1 plots the percentage of the benchmark dataset used in training on the x-axis and the CER on the y-axis. None of the architectures perform very well zero-shot - when trained only on synthetic data - and synthetic data generation is not this study's focus. On just 99 labeled table cells for Japanese and 21 labeled rows for LoCCA (the 5% train split), EffOCR's CER is only 5% (Japanese) and 7% (English), showing viable few shot performance. The other architectures remain unusable. EffOCR performs nearly as well using 20% or training data as using 70%, where it continues to outperform all other alternatives. This illustrates that its parsimonious architecture learns efficiently.

Supplementary Tables

	Horiz. Jap. Tables	**Vert. Japanese Tables**	**Vert. Jap. Prose**	**Chronicling America**
Train Lines	1309	898	459	291
Val Lines	280	192	98	62
Test Lines	281	193	100	64
Total	1870	1283	657	417
Train Chars	3089	3296	5832	7438
Val Chars	673	677	1063	1708
Test Chars	682	701	1111	1727
Total	4444	4674	8006	10873

Table S-1: This table reports the number of annotated lines and characters in the training, validation, and test sets of this study's four benchmarks.

		Feed Forward	Hard Neg.	No Synthetic Data		Encoder	
	EffOCR-C (Base)	Neural Net	Off	Recognizer	Localizer	Swin (Tiny)	ViT (Base)
Horizontal Japanese	**0.006**	0.006	0.041	0.594	0.009	0.009	0.010
Vertical Japanese (tables)	**0.007**	0.010	0.087	0.700	0.016	0.016	0.010
Vertical Japanese (prose)	0.030	0.038	0.076	0.788	0.032	0.036	**0.027**
Chronicling America	**0.023**	0.037	0.045	0.027	0.068	0.025	0.037

Table S-2: This table provides the character error rate. *Feed Forward Neural Net* models the recognizer as a classification problem with a feed forward neural network, *Hard Neg. Off* does not include hard negatives in recognizer training, *No Synthetic Data* turns off synthetic data training in the recognizer and localizer, respectively, and *Swin (Tiny)* and *ViT (Base)* are alternative vision encoders.

Supplementary Figures

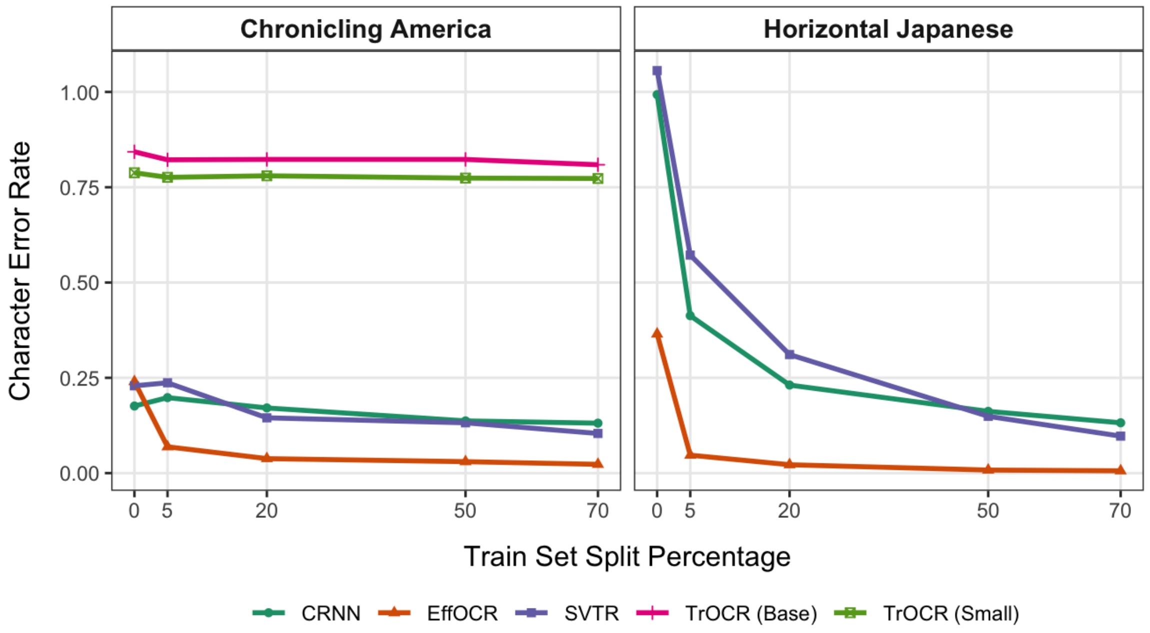


Figure S-1: **Sample Efficiency.** This figure plots the percentage of the benchmark dataset used in training against the character error rate, for different OCR model architectures.

References

1. Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anschel, Ron Slossberg, Shai Mazor, R Manmatha, and Pietro Perona. Sequence-to-sequence contrastive learning for text recognition. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15302–15312, 2021.
2. Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720, 2021.
3. Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34, 2021.
4. Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Mubarak Shah. Handwriting transformers. *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1086–1094, 2021.
5. Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.
6. Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5):1483–1498, 2019.
7. Jacob Carlson, Tom Bryan, and Melissa Dell. Effocr. `https://github.com/dell-research-harvard/effocr`, 2023.
8. Jacob Carlson, Tom Bryan, and Melissa Dell. Effsynth. `https://github.com/dell-research-harvard/effsynth`, 2023.
9. Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.
10. Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.
11. Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.
12. Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, Muriel Visani, and Jean-Philippe Moreux. Impact of ocr errors on the use of digital libraries: Towards a better access to information. In *Proceedings of the 17th ACM/IEEE Joint Conference on Digital Libraries*, JCDL '17, page 249–252. IEEE Press, 2017.
13. Theodore Cohen. *Remaking Japan: The American occupation as new deal*. Free Press, New York, 1987.

14. Mochikabu Kaisha Seiri Iinkai(Holding Company Liquidation Commission). *Nihon zaibatsu to sono kaitai (The Dissolution of Japan's Zaibatsu*, volume 2. Hara Shobo, Toyko, 1973.

15. Yongkun Du, Zhineng Chen, Caiyan Jia, Xiaoting Yin, Tianlun Zheng, Chenxia Li, Yuning Du, and Yu-Gang Jiang. Svtr: Scene text recognition with a single visual model. *arXiv preprint arXiv:2205.00159*, 2022.

16. Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval. *arXiv preprint arXiv:2102.05644*, 2021.

17. Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

18. He Guo, Xiameng Qin, Jiaming Liu, Junyu Han, Jingtuo Liu, and Errui Ding. Eaten: Entity-aware attention for single shot visual text extraction. *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 254–259, 2019.

19. Eleanor M Hadley. *Antitrust in Japan*. Princeton University Press, Princeton, NJ, 2015.

20. W Walker Hanlon and Brian Beach. Historical newspaper data: A researcher's guide and toolkit, 2022.

21. Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

22. Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online, June 2021. Association for Computational Linguistics.

23. Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

24. Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar. Icdar2019 competition on scanned receipt ocr and information extraction. *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1516–1520, 2019.

25. Jinji Koshinjo. *Jinji koshinroku*. Jinji Koshinjo, 1939.

26. Jinji Koshinjo. *Nihon shokuinrokj*. Jinji Koshinjo, 1954.

27. Glenn Jocher. YOLOv5 by Ultralytics, 5 2020.

28. Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

29. Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. The state and fate of linguistic diversity and inclusion in the nlp world. *arXiv preprint arXiv:2004.09095*, 2020.

30. Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

31. Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr github repository. `https://github.com/microsoft/unilm/tree/master/trocr`, 2021.

32. Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. *arXiv preprint arXiv:2109.10282*, 2021.

33. Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022.

34. Library of Congress. Chronicling America: Historic American Newspapers, 2022.

35. Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

36. Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

37. Lijun Lyu, Maria Koutraki, Martin Krickl, and Besnik Fetahu. Neural ocr post-hoc correction of historical corpora. *Transactions of the Association for Computational Linguistics*, 9:479–483, 05 2021.

38. Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.

39. Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. Pytorch metric learning, 2020.

40. Thi Tuyet Hai Nguyen, Adam Jatowt, Mickael Coustaty, and Antoine Doucet. Survey of post-ocr processing approaches. *ACM Comput. Surv.*, 54(6), jul 2021.

41. Joint Chiefs of Staff. Basic initial post surrender directive to supreme commander for the allied powers for the occupation and control of japan, jcs 1380/15, 1945.

42. Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

43. Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. Cord: a consolidated receipt dataset for post-ocr parsing. *Workshop on Document Intelligence at NeurIPS 2019*, 2019.

44. Shruti Rijhwani, Antonios Anastasopoulos, and Graham Neubig. Ocr post correction for endangered language texts. *arXiv preprint arXiv:2011.05402*, 2020.

45. Phillip Rust, Jonas F Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. Language modelling with pixels. *arXiv preprint arXiv:2207.06991*, 2022.

46. Zejiang Shen, Ruochen Zhang, Melissa Dell, Benjamin Charles Germain Lee, Jacob Carlson, and Weining Li. Layout parser. https://github.com/Layout-Parser/layout-parser, 2021.

47. Zejiang Shen, Ruochen Zhang, Melissa Dell, Benjamin Charles Germain Lee, Jacob Carlson, and Weining Li. Layoutparser: A unified toolkit for deep learning based document image analysis. *International Conference on Document Analysis and Recognition*, 12821, 2021.

48. Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

49. David A Smith, Ryan Cordell, and Abby Mullen. Computational methods for uncovering reprinted texts in antebellum newspapers. *American Literary History*, 27(3):E1–E15, 2015.

50. Guanglu Song, Yu Liu, and Xiaogang Wang. Revisiting the sibling head in object detector. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11563–11572, 2020.

51. Teikoku Koshinjo. *Teikoku Ginko Kaisha Yoroku*. Teikoku Koshinjo, 1957.

52. Narek Tumanyan, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Splicing vit features for semantic appearance transfer. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10748–10757, 2022.

53. Daniel van Strien., Kaspar Beelen., Mariona Coll Ardanuy., Kasra Hosseini., Barbara McGillivray., and Giovanni Colavizza. Assessing the impact of ocr quality on downstream nlp tasks. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence - Volume 1: ARTIDIGH,*, pages 484–496. INSTICC, SciTePress, 2020.

54. Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

55. Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics.

56. Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

57. Linjie Xing, Zhi Tian, Weilin Huang, and Matthew R Scott. Convolutional character networks. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9126–9136, 2019.