Applications of Machine Learning in Tabular Document Digitisation*

Christian M. Dahl† Torben S. D. Johansen† Emil N. Sørensen‡
Christian E. Westermann† Simon F. Wittrock†

December 29, 2021

Abstract

Data acquisition forms the primary step in all empirical research. The availability of data directly impacts the quality and extent of conclusions and insights. In particular, larger and more detailed datasets provide convincing answers even to complex research questions. The main problem is that "large and detailed" usually imply "costly and difficult", especially when the data medium is paper and books. Human operators and manual transcription have been the traditional approach for collecting historical data. We instead advocate the use of modern machine learning techniques to automate the digitisation and transcription process. We propose a customisable end-to-end transcription pipeline to perform layout classification, table segmentation, and transcribe handwritten text that is suitable for tabular data, as is common in, e.g., census lists and birth and death records. We showcase our pipeline through two applications: The first demonstrates that unsupervised layout classification applied to raw scans of nurse journals can be used to obtain valuable insights into an extended nurse home visiting program. The second application uses attention-based neural networks for handwritten text recognition to transcribe age and birth and death dates and includes a comparison to automated transcription using Transkribus in the regime of tabular data. We describe each step in our pipeline and provide implementation insights.

*Acknowledgements: We thank Peter Sandholdt Jensen, Joseph Price, Michael Rosholm, an editor, and three anonymous referees for valuable comments and suggestions that have improved the manuscript substantially. We also thank Søren Poder for his expertise on digitisation of historical documents. We gratefully acknowledge support from Rigsarkivet (Danish National Archive) and Aarhus Stadsarkiv (Aarhus City Archive) who have supplied large amounts of scanned source material. We also gratefully acknowledge support from DFF who has funded the research project "Inside the black box of welfare state expansion: Early-life health policies, parental investments and socio-economic and health trajectories" (grant 8106-00003B) with PI Miriam Wüst. Data (excluding nurse records) and code are available on request.

†Department of Economics, University of Southern Denmark

‡School of Economics, University of Bristol, corresponding author, e.sorensen@bristol.ac.uk

1. INTRODUCTION

Big data and machine learning (ML) receive significant attention in the historical and digital humanities literature, especially as archives and historians have realised how the rapid growth of data can challenge the scalability, speed, and costs of traditional digitisation methods. In economic history, Gutmann et al. (2018) argue that large collections of historical documents are important examples of big data and that these call for development of new and efficient methods.

Paper-based sources are often manually digitised by researchers or crowdsourcing platforms. However, manual digitisation becomes infeasible as the volume of data grows. This is especially true for structured and tabular data which have high information density and exist in massive quantities at the population level.[1] Examples include census lists, birth and death records, church records, medical records, school grade sheets, logbooks, weather measurements, and school diaries. While such large datasets provide compelling opportunities to study historical phenomena, especially long-run outcomes and intergenerational transmission, see, e.g., Gutmann et al. (2018), Feigenbaum (2018), and Abramitzky et al. (2021), they also pose significant challenges to the transcription process due to their scale.

ML is a potential solution to this problem as it can automate the digitisation and transcription processes. The main gains of automated transcription by ML is reproducibility, cost efficiency, and scale. This has led to a focus on new ML-based archiving and digitisation methods that are designed for big data, see, e.g., Rosenzweig (2003), Colavizza et al. (2019), and Moss et al. (2018). Relevant to our setting, several methods have been developed to automate the transcription of scanned documents. Traditional optical character recognition (OCR) has been widely applied for documents with machine written text.[2] Handwritten text recognition (HTR) has also been used with success, although it often requires more

[1]For a description of structured data, see, e.g., https://www.archives.gov/records-mgmt/policy/transfer-guidance-tables.html#structuredata

[2]Common OCR tools include Google Vision AI, Amazon Textract, Adobe Acrobat, Abby FineReader, and Tesseract.

complex ML methods to achieve adequate performance in documents with varying layout and script. 'Transkribus' (Muehlberger et al., 2019) and 'Monk' (Schomaker, 2020) are publicly available projects that focus on transcription of historical manuscripts with streams of handwritten text.[3] To use Transkribus, a sufficiently large number of pages must be manually annotated with respect to layout, baselines, and text to establish a ground truth that can guide the transcription.[4] Transkribus then uses ML to perform layout analysis, line segmentation, and transcription. Monk works similarly but improves performance when documents are heterogeneous in layout and handwriting style, see Weber et al. (2018) and Schomaker (2020).

Both OCR and HTR methods have significantly contributed to the transcription of books and long-form manuscripts like newspaper articles, see, e.g., The Australian Newspapers Digitization Program by the Australian National Library.[5] However, structured and tabular documents pose a substantially different challenge. Tabular data are characterised by a large number of cells that are organised into tables. Such tables have high information density, and due to the cell structure, there is substantial within-table variation in the locations and baselines of the text. These complexities make it difficult to apply off-the-shelf tools. Also, apart from transcribing the text, we also need to analyse the table structure to organise the transcribed information and use it in downstream analyses. Technologies such as Transkribus and Monk are not specifically designed to transcribe tabular data and they require manual adaptations to work on complex tables, see, e.g., Muehlberger et al. (2019).[6] While there has been work to adapt Transkribus to table transcription, the current approaches are not fully automatic and they entail a significant manual workload related to segmentation and drawing of baselines.[7] When the table complexity is high and the document count is large,

[3]Official website of Transkribus: https://readcoop.eu/transkribus/. Official website of Monk: http://monk.hpc.rug.nl/

[4]Approximately 75 pages according to Muehlberger et al. (2019)

[5]An overview of this project is available at http://www.nla.gov.au/ndp/project_details/

[6]Both Transkribus and Monk also facilitate keyword spotting but, when the end goal is to build complete databases, this is not easily adaptable to transcribe, e.g., dates

[7]The official website of Transkribus discusses this: https://readcoop.eu/transkribus/howto/how-to-work-with-tables-in-transkribus/. Also, Transkribus and databases are discussed by https:

even minor manual input per document will drastically increase the overall workload. In such cases, fully automated transcription is preferable. These considerations leave significant unsolved challenges in the transcription of large volume tabular data. In this work, we focus specifically on tabular data and we propose a custom end-to-end ML pipeline for automated transcription of tabular documents with handwritten text. As we demonstrate, our pipeline is particularly well suited to transcribe large collections of documents containing population data such as mortality statistics, names, dates, cause of death, occupation, place of residence, sex, civil status, etc.

We test and illustrate our pipeline on two large collections of historical documents. In the first case study, we demonstrate that we can use layout analysis, 'step one' of our pipeline, in itself to collect data. We apply layout classification to a complex tabular dataset of nurse records. We also discuss how the ML method gives additional insights on treatment assignment that complement, and expand on, the results from an intention to treat analysis, without requiring additional manual transcription of the source data. The second case study extends this example and shows that, after the initial layout classification, we can use table segmentation and handwritten text recognition to transcribe handwritten information inside the tabular document. We also show that, based on the raw scans, the pipeline can automatically collect lifespan data that are directly useful in modeling mortality risk. In this application, we consider all three steps of the pipeline, and apply the methods to a subset of a large collection of death certificates. The death certificates are interesting as they represent a realistic example of a large structured dataset, the images are non-trivial to transcribe (significant variation in layout and script), the collection is large, and it is available online. Finally, we also compare the ML approach to traditional crowdsourcing and Transkribus, and show that ML can significantly reduce the transcription burden for tabular documents while performing on-par with crowdsourced transcriptions and better than transcriptions from Transkribus. In particular, we find that accuracy of our transcriptions are close to

`//oliverdunnresearch.files.wordpress.com/2019/04/transkribus-and-database-creation.pdf`.

those from crowdsourcing, and that the statistical distribution of the transcribed data is roughly equivalent to that of the ground truth data. This is especially important if the data are to be used in downstream statistical models.

The rest of the paper is structured as follows. Section 2 provides an overview of our ML pipeline for tabular documents. Section 3 discusses two case studies where we test the pipeline. Section 4 compares the results from the ML pipeline with data obtained from crowdsourcing and Transkribus. Section 5 concludes. The Technical Appendix documents the implementation details of our approach, so it can be used by other researchers. Our datasets and code are available on request.

2. PIPELINE

We now discuss the structure of our ML pipeline. The goal is to transform raw scans of tabular documents into datasets. In general, such scans are heterogeneous in resolution, table structure, and script. As a consequence, and to maximise the generalisation capabilities, our ML pipeline is split into three sequential components: (1) 'layout classification' sorts the documents based on layout, (2) 'table segmentation' extracts the cells of interest from the source images, and (3) 'transcription' transcribes the extracted cell images. Each component is responsible for a specific task and can be adapted to the problem at hand. Figure 1 illustrates the pipeline. We emphasise that the pipeline is highly customisable and that the downstream research question, and statistical model, should dictate the choice of individual components in the pipeline. For example, if our research question focuses on mortality, we can limit our attention to cells containing death dates and we can tailor our transcription model to dates. Such constraints can accelerate the learning process.

The inputs to the pipeline are scanned documents stored digitally as image files. These images consist of coloured dots called pixels. Each pixel is characterised by a location and a colour, and stacking a certain number of pixels horizontally and vertically forms an image. Thus, we can consider an image of height h and width w to be a $h \times w$ matrix where each

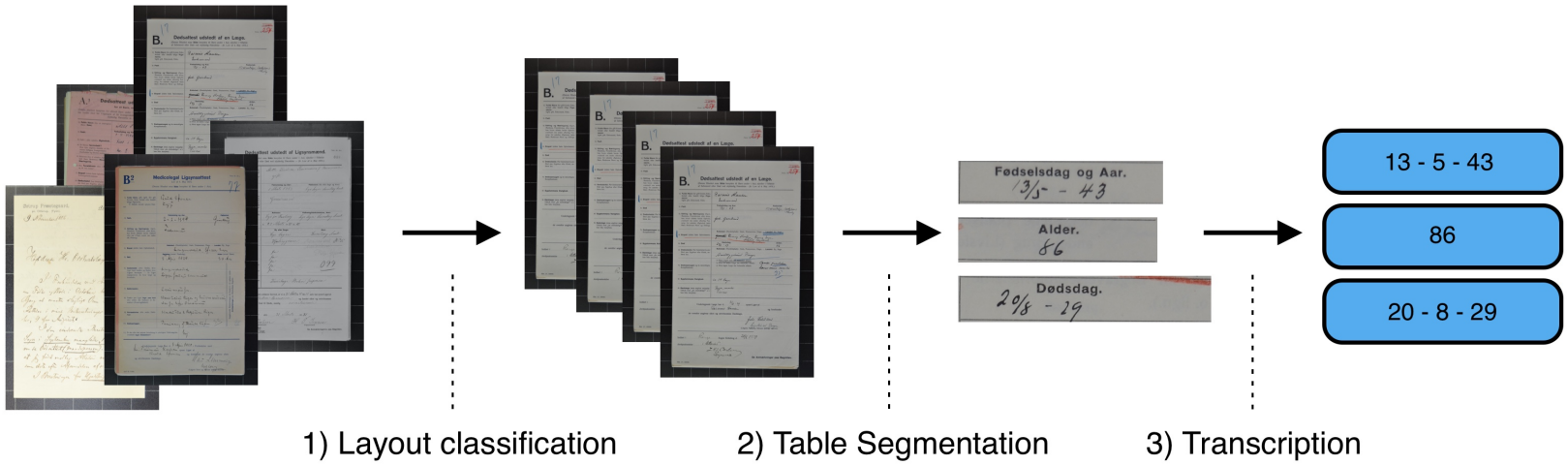

Figure 1: The three steps in the ML data collection pipeline. (1) the collection of source images is sorted by layout. (2) the form in the document of interest is segmented into field images. (3) each field image is transcribed into a digital string.

entry corresponds to a single pixel. The objective of the pipeline is to learn a mapping from the image matrix into a representation suitable for statistical analysis or storage in a database. However, this is both complex and challenging as the image matrix can be of very high dimension, which calls for specialised statistical models.

The first step of the pipeline is layout classification. Layout classification refers to the process of organising a document collection by common layout structures, e.g., a common table, a heading, or pre-printed landmarks. The layout type is important as the table segmentation relies on a pre-defined template that must match with the pre-printed structure in the image. Due to variation in layout across a given document collection, we need to construct one template for each layout type, and when we fit a template to a document, we need the template and layout types to match. This is straightforward if the documents are sorted according to layout as all images of a given layout type will share the same template. Our pipeline allows for different layout classification models and we showcase both an unsupervised and supervised approach in, respectively, Section 3.1 and Section 3.2. The unsupervised method clusters the documents based on visual descriptors extracted by a neural network. However, while its performance is impressive in our first case study, it does not generalise to more heterogeneous documents. This motivates a supervised approach in our second case study. In the supervised setting, we use the Bag-of-Words (BoW) method which was originally developed for classifying chunks of text (Murphy, 2012, p. 87). It has since been succesfully applied in the field of computer vision, see, e.g., Csurka et al. (2004) and Sivic and Zisserman (2009). BoW compares documents based on the frequency of certain clusters of key points, so-called 'visual words'. Examples of visual words are the corners of a table, or a particular document heading. The key points are based on Speeded-Up Robust Features (SURF) (Bay et al., 2008) as this has historically been the common choice, see, e.g., Csurka et al. (2004). As a result, every document will have an associated histogram that describes its distribution of these visual words. We then train a model to classify a given document into its respective layout group based on its histogram. The classification

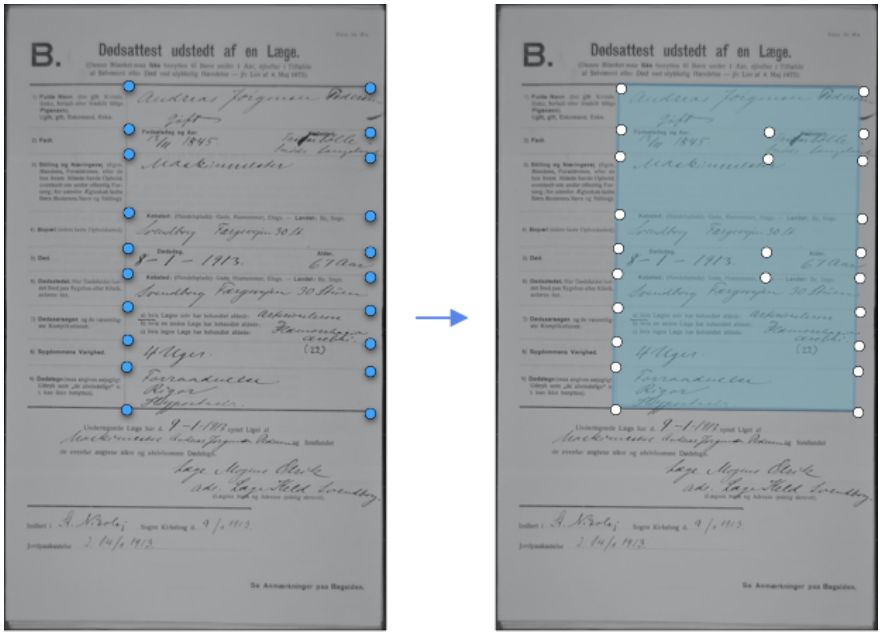

Figure 2: Left: Template points defined at line intersections and endpoints. Right: Overlay with additional points to isolate fields of interest.

model can be of any type, but we use support vector machines with radial basis kernels. We provide additional details on layout classification in the Technical Appendix A.2.2.

The second step of the pipeline is table segmentation where we extract images corresponding to each field (or cell) in a given table in the source image. Coüasnon and Lemaitre (2014) provide a general introduction to this topic. For a given classified group of document images, we need to choose a reference or 'template' document. The template can be any well-scanned document in the respective layout group. We note the coordinates of line intersections and line endpoints and refer to them as the template key points. We next construct an 'overlay', which is the set of rectangles that encloses each field of interest and where each rectangle is represented by the coordinates of its four corners. This is done using the same document that provided the template. Constructing the template and overlay is an operation that has to be carried out only once per document group. Figure 2 illustrates the template and the corresponding overlay for a Danish death certificate, see also Section 3.2. With the template and overlay in place, we process the remaining documents of the respective group, which we refer to as the 'target' images. For each target image we need to locate the same

key points as those defined in the template. This is done with standard computer vision operations that identify the vertical and horizontal lines of the table structure, see Figure A.4 for a visual example. Based on the identified lines, we use their intersections and endpoints to find the same key points that were defined in the template. What remains is to find the transformation that maps the key points of the target image to their respective counterparts in the template. To do this, we use Coherent Point Drift (CPD) which iteratively aligns the two point sets (Myronenko and Song, 2010). After applying CPD, we obtain translation and transformation parameters that, when applied to the target key points, aligns them with those of the template. We can then extract each field using the transformed coordinates. There are also other methods that could be adapted to segment the documents, e.g., Li et al. (2019) and Clinchant et al. (2018), and these could seamlessly be integrated into our pipeline. However, we do not consider this here. We provide details on table segmentation in the Technical Appendix A.2.3

The final step of the pipeline transcribes the extracted fields. Transcription is the process of converting an image of text into a string representation. We use an attention-based neural network, suggested by Xu et al. (2015) for image captioning, and adapt it to transcription of handwritten text. The model by Xu et al. (2015) applies to tasks with image inputs and sequence outputs, which is also what characterises the transcription problem where the output is a sequence of characters. An advantage of the attention approach is that the model requires only rough segmentation at the field level and does not rely on, e.g., text baselines. The model processes either characters or words and we limit the set of possible characters or words based on the task at hand. This set could either be the alphabet in the case of name transcription or digits for age transcription. We provide additional details in the Technical Appendix A.2.4.

3. APPLICATIONS

3.1. Nurse records and infant care

In economics, there is a large literature on the relationship between early-life conditions and later-life outcomes, see, e.g., Almond and Currie (2011a), Almond and Currie (2011b), Almond, Currie, and Duque (2018), and Hoehn-Velasco (2021). This is commonly studied in a regression framework where data on individuals are collected at birth, or during childhood, and linked to outcomes in adulthood. These studies inherently require long-run historical datasets that cover the individuals' lifespans. In the following, we exemplify such a study by considering the impact of early-life care on outcomes in adulthood.

To estimate a causal effect of infant care, we need an assignment of each individual to a treatment or control group. In the social sciences, we often infer treatment assignment based on an intervention or policy that (quasi) randomly has assigned each individual to treatment or control, e.g., Angrist and Krueger (1991), Angrist, Imbens, et al. (1996) and Imbens and Rubin (1997). This section considers a policy where a subset of infants was made eligible to participate in an expanded care programme. The participants in the programme received additional home visits from nurses. Enrolment in the programme was governed by the date of birth. Individuals born in the first three days of each month were eligible to receive additional monitoring. The details of approximately $95,000$ infants (whether enrolled or not) were collected in journals kept by the health care system. The journals have previously been described and used by Biering-Sørensen et al. (1980), and Bjerregaard et al. (2014) have manually transcribed a small subset of the contents to study birth weight and breastfeeding. The infants who received additional monitoring have a specific follow-up table in their journal only if the monitoring took place, i.e. the presence of the table is decided by actual treatment, not eligibility. The journals have been scanned and are available as digital images.[8] While parts of the journals have previously been digitised, the presence of

[8]The journals have been made available through the DFF funded research project "Inside the black box of welfare state expansion: Early-life health policies, parental investments, and socio-economic and health

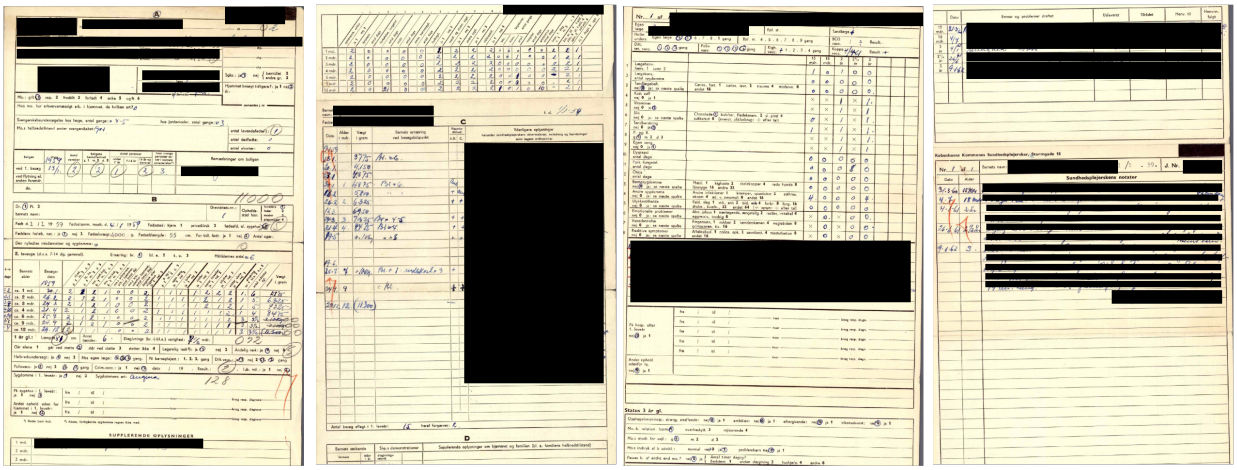

Figure 3: Example of a typical nurse journal. The third page shows the treatment table. We have censored sensitive information.

the treatment table was not recorded. Figure 3 illustrates the pages in a typical journal.

In the following, we use the layout classification step of our ML pipeline to detect whether an infant received extended care, i.e. was treated. This is to illustrate that 'step one' of the pipeline can, in isolation, collect important data for applied research, and that transcription is not always needed after the initial layout analysis. It can save both time and complexity if the digitisation process is adapted to the downstream research question, and one strength of our pipeline is that it allows for such adaptations. To detect whether an infant was treated, our ML model analyses the layout of each journal page and identifies the group of children that received follow-up care. We compare this ML-based detection to an intention to treat indicator inferred from the three-day policy and find that there is so-called non-compliance (Angrist, Imbens, et al., 1996). Our dataset contains 95,323 journals with a total page count of 261,926. The page count and order vary between journals. This implies that all pages in all journals have to be reviewed to identify the treated. Since the treated individuals can be identified by the presence of a particular page in their journal, we can use layout classification to detect treatment. If a page in their journal is classified as having the treatment table layout, then the individual is classified as treated. We did not have access

trajectories" (grant 8106-00003B) with Miriam Wüst as PI.

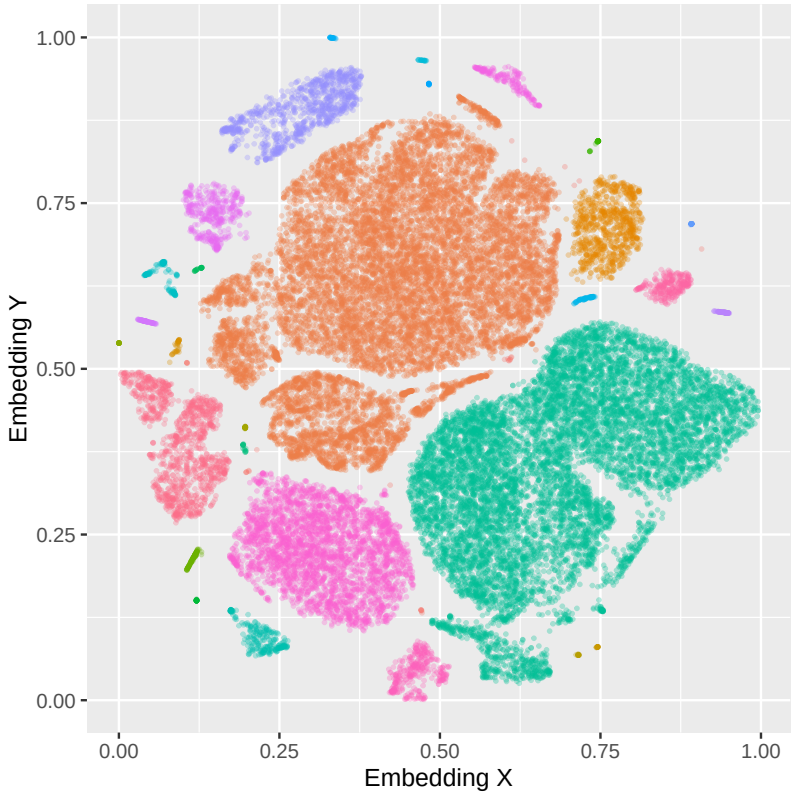

Figure 4: Visualisation of the feature space of the journal pages. Each point represents a journal page and the colours correspond to the labels assigned by the clustering algorithm. Pages with similar layout cluster together. The embeddings have been subsampled to reduce cluttering, so only 30,000 randomly sampled embeddings are displayed. There is a total of 37 clusters which are manually annotated. The treatment pages are contained in four clusters.

to a labelled dataset to train a supervised classifier for the treatment page – as will often be the case in practice. Thus, we pursue an unsupervised approach where we rely only on the scanned images without labels. The technical details are described in Appendix A.1. Note that we still need to manually construct an evaluation dataset to probe the performance of the applied method. We will show an example of a supervised classifier for the same purposes in Section 3.2, where training data is available.

3.1.1. Results

We now discuss the results from applying 'step one' of our pipeline to the nurse records. The layout classification method clusters pages together according to their visual appearance. To

	Policy detection (ITT)	ML detection
Treated	8,596	5,735
- Born 1st-3rd	8,596	4,901
- Born 4th-31st	0	397
Non-compliers	-	4,092

Table 1: Treatment indicator. Policy assignment is based on an official assignment rule which offered all children born in the first three days of each month to enroll in the nurse visiting programme. The ML assignment is based on the machine learning model and bases assignment on the presence of the treatment page in the journals. This allows for assessment of compliance in addition to the intention-to-treat effect, i.e. the date-of-birth assignment mechanism.

efficiently describe the visual information of the nurse journals, we use a neural network to construct a low dimensional representation of each journal page. We then apply a density-based clustering algorithm to the output of this neural network to recover the different layout types. In Figure 4 we illustrate this process by showing how the pages separate into different clusters based on their visual appearance. Each point in Figure 4 represents a page in a journal and the points are coloured according to their assigned cluster. A clear structure is evident. There are 37 clusters which we manually review and annotate according to their contents. Annotation is carried out by randomly sampling 10 pages from each cluster and assigning a label for the whole cluster based on the contents of these pages. This amounts to 370 journal pages that need manual review (out of $261,926$). This procedure shows that the treatment pages are contained in four distinct clusters. We extract all pages residing in these clusters and classify the underlying individuals as treated, i.e. an individual is treated if any page in their journal belongs to one of the four treatment page clusters.

To evaluate the unsupervised procedure, we manually construct a ground truth evaluation dataset by reviewing all $10,914$ pages of $4,000$ randomly selected journals. For each journal, we record the presence of the treatment table. We review each dataset twice and find 234 treated journals. Table A.1 shows a confusion matrix for the ML treatment detection in the evaluation sample. All 234 treated and $3,766$ untreated individuals are correctly classified.

Due to class imbalance, the classifier could obtain an accuracy of $3,766/4,000 \approx 94.15\%$ by simply assigning everyone as non-treated.[9] Thus, more informative performance measures are precision and recall (Murphy, 2012, p. 184–185). In our context, precision is the fraction of individuals correctly predicted as treated out of the total number of individuals predicted as treated', while recall is 'the fraction of individuals correctly predicted as treated out of the total number of treated individuals' . In the ground truth sample, the unsupervised method has precision and recall equal to unity.

Apart from the performance of the classifier itself, the results from the layout detection give insights on treatment assignment. Table 1 shows that not all eligible individuals received the follow-up visits. $8,596$ children were eligible but only $4,901$ individuals born in the three-day eligibility period received a visit.[10] This is a treatment uptake of 57.01%. In addition, there is a group of 397 children born outside the intervention days that still receive treatment even though the policy assigns them as controls.[11] This reveals an issue with non-compliance ($8,596 - 4,901 + 397 = 4,092$ non-compliers). Non-compliers are individuals that do not comply with their assignment to treatment (or control), e.g. children born between the 1st and the 3rd that do not receive follow-up care. For statistical reasons, this is important when studying the effects of extended care in a regression framework, see, e.g., Angrist, Imbens, et al. (1996). Without applying our ML pipeline, these details on the intervention would not have been discovered. That is, unless all $261,926$ pages were manually reviewed.

[9] Actually, the class imbalance is even more severe. The ML model classifies pages and in the evaluation set only $234/10,914 \approx 2.14\%$ pages contain the treatment table. In light of this, the high recall of the model is especially satisfying.

[10] Although $8,596$ individuals born between the 1st and the 3rd appears low when considering the sample size of roughly $95,000$ children, we observe birth date for only $86,790$ children. Hence, 9.90% of these children are born between the 1st and the 3rd.

[11] A search for more information on these cases revealed these were mostly twins, which were assigned follow-up regardless of whether they were born between the 1st and the 3rd.

3.2. Death certificates and mortality

Denmark introduced the national use of death certificates in 1832.[12] A death certificate documents the death of a single individual. They are used to study a number of historical, economic, and health questions at the individual and population levels (Acuña-Soto, Breman, et al., 2012). Importantly, the certificates contain individual-level lifespan and cause-of-death data, which could be used to model mortality risk (Högberg and Wall, 1986). Potentially, they can also be linked to other registries and datasets to model the relationship between mortality and external conditions, e.g., Berg et al. (2006) and Bruckner et al. (2014). The information from the death certificates also play central roles in studies of place-of-death (Cohen et al., 2007) and pandemics (Simonsen et al., 2011; Gill and DeJoseph, 2020).

The death certificates are an important example of historical big data as they are hardly feasible to manually transcribe. It is estimated that over 3 million Danish death certificates exist and each of them contains 10-12 fields, see Figure 5 for an example.[13] This equals millions of individual fields to transcribe. However, as we will illustrate, our ML pipeline is well suited for efficient transcription at this scale. In this case study, we work on a subsample of approximately $250,000$ death certificates that were collected across multiple years and locations. The certificates contain pre-printed forms that are filled in by hand. The forms change over time and several subtypes are used to distinguish between deceased infants, suicides, accidents, etc. These are illustrated in Figure A.3 in the Appendix. The de-centralised scanning of the documents, by multiple archives with different volunteers and equipment, has resulted in substantial variation in scan quality. Unlike the first case study, we use all three steps of our pipeline to illustrate a fully automated approach where the pipeline transforms raw scans into transcribed lifespan data. For the sake of illustration, and to limit the complexity, we focus only on transcription of dates (birth and death) and age.

[12] See the description at the Danish National Archive (in Danish) https://www.sa.dk/da/hjaelp-og-vejledning/rigsarkivets-online-vejledninger/doedsattester-kom-godt-i-gang/

[13] The fields on each certificate are – with some variation – name, marital status, birth date, death date, age, occupation, birth location, death location, cause of death, duration of disease, and clinical signs of death.

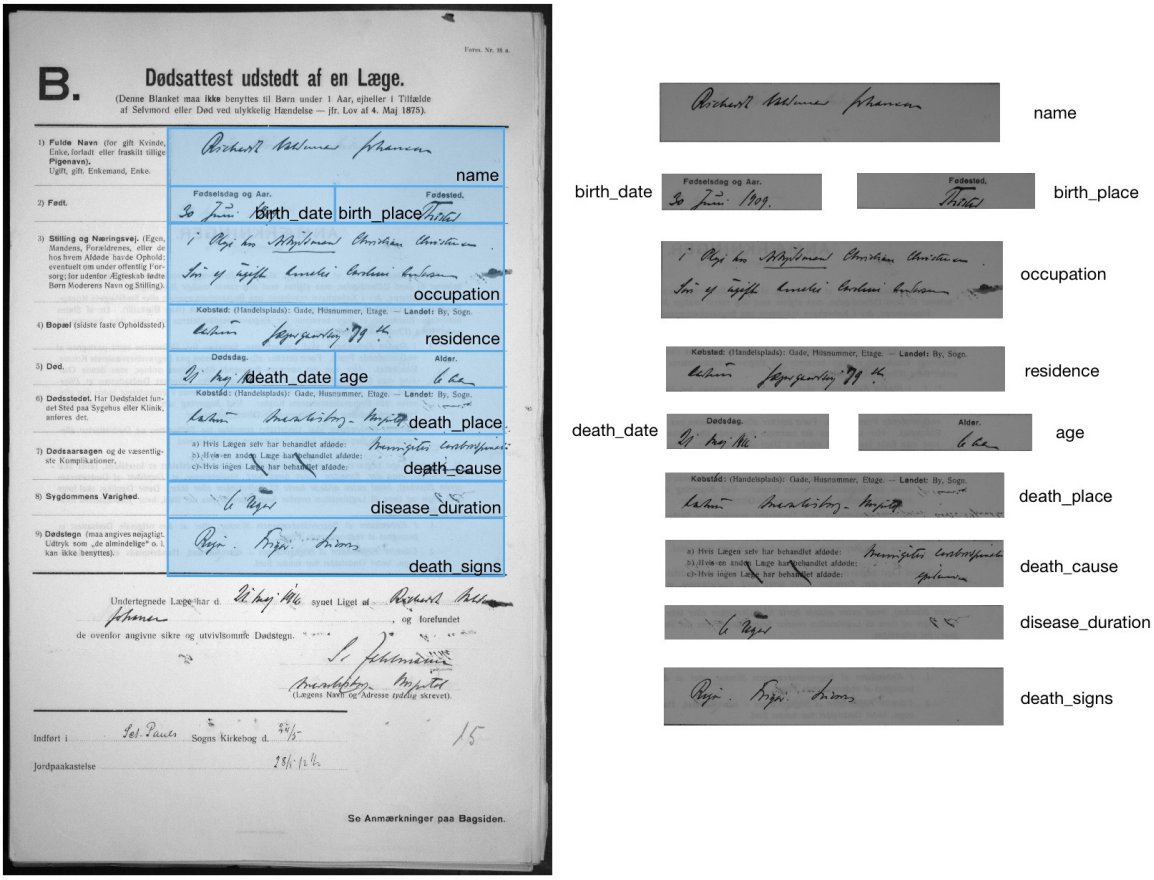

Figure 5: The fields on a type B death certificate.

These variables are important in studies of mortality and they have a well-defined dictionary consisting purely of numbers 0-9 and months (January-December). Also, these fields allow for an internal consistency check as the difference between an individual's birth and death dates must match the individual's age. The fields are still challenging to transcribe as they are handwritten and vary significantly in script and format, in part due to the presence of multiple authors, see Figures 6 and 7. We limit our attention to the type B death certificates, as the pre-printed form has clearly delineated fields and our collection has a large proportion of type B certificates.[14] Figure 5 shows a type B certificate where we highlight the relevant fields, including the fields for age and birth and death dates.

We apply the pipeline to the type B death certificates to transcribe age and birth and

[14]Our layout model – to be discussed later – predicts that 44,903 out of 250,000 certificates are type B.

	ML detection				
Ground truth	Empty	Other	A	B	
Empty	13	1	0	0	14
Other	2	1,535	0	0	1,537
A	0	0	109	0	109
B	0	0	0	524	524
	15	1,536	109	524	

Table 2: Confusion matrix for the BoW layout classification. The frequencies are based on a randomly sampled and manually reviewed evaluation set of 2,184 death certificates. Death certificates are classified into four classes: Empty, A, B and Other. In this application we are only interested in the type B certificate.

death dates. The pipeline uses BoW for layout classification, CPD for segmentation, and attention-based neural networks for transcription. In this process, we rely on several ground truth datasets that we describe in Appendix A.2. As the table segmentation method does not need training there is no dataset for this step. We construct the evaluation and training datasets by manually transcribing a random sample of images from the death certificates. We verify each image twice and discard images with segmentation errors. Importantly, there is no overlap between the training and evaluation datasets.

3.2.1. Results

Layout classification. We train the layout classifier on a dataset containing 7,000 randomly sampled death certificates and their ground truth layout type. We consider four distinct layout classes: (1) type A certificates, (2) type B certificates, (3) all other certificates, and (4) empty pages. We evaluate the classifier on 2,184 certificates. Table 2 shows the confusion matrix. The classifier performs well with only two false positives and one false negative, and the class-wise precision and recall for type B certificates are unity. After evaluation, we use the classifier to predict the layout class of the 250,000 death certificates and extract 44,903 certificates that are classified as type B.

Table Segmentation. Our table segmentation method does not need training data except for an initial template that is manually drawn to match the form in the document. We

apply the segmentation method to the 44,903 type B certificates, extracted in the layout classification step, and we segment the fields of interest. The segmentation is not without errors. Segmentation errors typically implies that parts of the image fields may be left out from the segmented images. Naturally, this affects the final transcription accuracy. We discuss this issue further in Section 4.

Transcription. We evaluate the performance of the transcription models using two metrics. One relates to the average accuracy across individual components (called tokens) in the sequence of numbers and characters, denoted TA, and the other to the accuracy of complete sequences, denoted SA.[15] Specifically, SA_m is the proportion of correctly predicted sequences when m mistakes are allowed in the sequence. The token and sequence accuracies are closely related to the character and word accuracies in the HTR literature, see, e.g., Graves, Liwicki, et al. (2008).[16] This section focuses on the performance of the ML transcription models in isolation, so all results are conditional on the transcription models receiving perfectly segmented images. The trained transcription models are used to predict the images in the date and age evaluation datasets using beam search. Prior to prediction, the images are standardised to zero mean and unit variance using the mean and variance estimated from the entire evaluation dataset.[17] No other pre-processing is applied. Transcription of a single image happens in less than 75ms.

Table 3 shows the performance of the two transcription models – for date and age respectively – on their corresponding evaluation sets. We present results both for training with and without augmentation of the training dataset, where augmentation is the process of slightly altering the images to generate more data. The token accuracy (TA) for dates is 97.9% with augmentation and 92.2% without. For ages the TA is 98.5% with augmentation and 96.6% without. These are the average accuracies of predicting a single token correctly. Figures 6-7 show random samples of incorrect and correct dates as predicted by the ML date

[15] Examples of tokens and sequences are provided in Appendix A.2.4.

[16] The definitions of TA and SA_m are given by Equations A.1–A.2 in Appendix A.2.4.

[17] This is common practice and as straightforward as it sounds. We take the average and standard deviation over all pixels in all images, and then respectively subtract and divide each pixel by these values.

	Date				Age		
	TA	SA_0	SA_1	SA_2	TA	SA_0	SA_1
Real	.922	.661	.933	.981	.966	.936	.988
	(.008)	(.015)	(.008)	(.004)	(.006)	(.008)	(.003)
Real w. augmentation	.979	.905	.989	.991	.985	.972	.999
	(.005)	(.009)	(.003)	(.003)	(.004)	(.005)	(.001)

Table 3: Token (TA) and Sequence (SA) Accuracy on ground truth evaluation set. Standard errors are given below each accuracy rate. The first row uses only ground truth training samples, while the second row shows the result when training is conducted on the augmented dataset. The date training set contains $11,320$ samples, the evaluation set contains $1,000$ samples. The age training set contains $11,072$ samples, the evaluation set contains $1,000$ samples. The dates contain both birth and death dates. Note that these are not end-to-end accuracy rates, so they do not factor in the performance of the table segmentation (i.e. segmentations that obscures the written information have been discarded). The accuracy excludes the `<Start>` token but includes the `<End>` token (the `<Start>` token is forced in the network, so it will always be present).

transcription model.

In Table 3, the zero-error sequence accuracies SA_0 are significantly lower than the token accuracies for both models. Under augmentation, the date model achieves an SA_0 of 90.5% while the age model has an SA_0 of 97.2%. In the context of US censuses, Nion et al. (2013) transcribed age at a sequence accuracy of approximately 85% using convolutional neural networks.

Notice that a sequence prediction is only correct if all tokens are predicted correctly. This implies correctness of 11 tokens for dates and 4 tokens for ages (including `<Start>` and `<End>` markers). Hence it is no surprise that the zero-error sequence accuracy is much higher for age than date. Also, the variation in dates is larger compared to ages with respect to both format and combination of digits and characters.

It is apparent from Table 3 that an increase in the number of allowed mistakes per sequence improves the accuracy significantly. For example, if we allow one mistake (i.e. one substitution) in the date sequence, the one-error sequence accuracy is 98.9%. Under some circumstances, e.g. linking, it might be acceptable with a certain number of mistakes in the

Figure 6: Random sample of incorrect predictions from the date transcription model. The label in the upper right corner displays the model prediction in the format day–month–year.

Figure 7: Random sample of correct predictions from the date transcription model. The label in the upper right corner displays the model prediction in the format day–month–year.

sequence. Also, in statistical models, the transcription errors might not matter unless they depend systematically on the transcribed information.

Notice also the difference between using augmentation and not in the first and second row of Table 3. The significant differences are due to the small training datasets of 11,630 dates and 11,072 ages. If our training datasets were larger, the payoff from augmentation would be smaller. However, the differences display the benefits of augmentation to boost performance in smaller training datasets.

4. COMPARISON OF ML, CROWDSOURCING, AND TRANSKRIBUS

To gauge the performance of our ML pipeline, we compare the ML predictions to transcriptions from crowdsourcing and Transkribus. The usefulness of this comparison is twofold. First, we can evaluate if the ML pipeline performs on-par with crowdsourcing and Transkribus. Second, using the large crowdsourced dataset, we can estimate the end-to-end performance of the ML pipeline including errors of segmentation, transcription, and, to an extent, layout classification.[18] The crowdsourced dataset is freely available online from the Danish National Archive, and anyone can contribute to the dataset through their website.[19]

Our own evaluation dataset – as used in Table 3 – was manually reviewed to exclude (1) images where segmentation errors obscured the text in the image, and (2) where the image belonged to a document of the wrong layout type (i.e. anything other than type B death certificates). Evaluating the model on this dataset provides a clean measure of the transcription model in isolation. However, it does not give any insights on the transcription performance when the model might receive a badly segmented image. The crowdsourced dataset is different in this respect as it is constructed by humans looking at the raw document, finding the relevant field, and transcribing the text. This implies that the crowdsourced

[18] This does not give us a good measure of the recall of the layout classifications as we only use certificates that are in both datasets. To see this, note that there might be type B certificates that (1) have not been detected by our ML model and (2) are not in the crowdsourced dataset. These certificates will be missed here.

[19] See (in Danish) https://bit.ly/2VnFLzb

	Date		Day		Month		Year	
	ML [1]	Crowd [2]	ML [3]	Crowd [4]	ML [5]	Crowd [6]	ML [7]	Crowd [8]
SA_0	.905	.963	.960	.983	.970	.987	.972	.988
	(.009)	(.004)	(.006)	(.002)	(.005)	(.002)	(.005)	(.002)

Table 4: Zero-error Sequence Accuracy SA_0 for the ML (pipeline) data model (trained with augmentation) and crowdsourcing both evaluated on our date ground truth datasets. Second row contains the standard error of the accuracy rate. The evaluation set for the ML (pipeline) model consists of 1,000 samples, while the evaluation set for the crowdsourced predictions consists of 2,864 samples as we can pool both our training and evaluation ground truth datasets in this case. The training and evaluation sets have been twice manually reviewed and dates that are unreadable due to bad segmentation have been removed. Age is not directly transcribed in the crowdsourced dataset and hence excluded here. Columns 1-2 are sequence accuracies for the whole date, while columns 3-8 are sequence accuracies on the individual components of the date (day, month, and year). Note that the comparison takes into account common date formatting, e.g. that the dates 01-10-2000 and 1-10-2000 convey the same point in time.

transcriptions cannot be impacted by segmentation or layout classification errors. By running the entire pipeline on the raw images, and comparing the final transcription output to the crowdsourced transcription, we can get a measure of the overall performance of the pipeline in practice. Of course, this relies on the assumption that the crowdsourced data are perfectly transcribed. Thus, to get a baseline indication of the quality of the crowdsourced dataset, we compare it against our ground truth training and evaluation sets (overlap of 2,864 documents) and find that the dates are identical in 96.30% of the cases, see Table 4. For comparison, the performance of the transcription model on the evaluation set (i.e. perfectly segmented images) is 90.5%. If we look at the individual components of the date, the ML performance is 96%, 97%, and 97.2% on days, months, and years, respectively. For crowdsourcing, the corresponding accuracy rates are 98.3%, 98.7%, and 98.8%. Thus, on the individual date components, the accuracies of crowdsourcing and ML are fairly close, although statistically different. However, this neglects the impact of the layout classification and segmentation steps in the pipeline.

Next, we evaluate the ML transcriptions by using the crowdsourced transcriptions as

ground truth. The crowdsourced transcriptions overlap with our sample for 23,263 documents – each containing a birth and death date for a total of 46,526 dates – and we filter out any overlap with the training sample used to train the ML model. Note that the crowdsourced dataset does not contain transcriptions of age. The dates predicted by the ML model and crowdsourcing are identical in 83.66% of the cases and for 89.96% of the dates the difference is less than one calendar year.[20] This is a substantial difference compared to Table 3 where the ML sequence accuracy was 90.5%. As elaborated above, the performance difference relative to Table 3 stems from two sources: (1) noise in the crowdsourced dataset and (2) the other pipeline steps prior to transcription. Thus, unless (1) is large, this gives an approximation to the end-to-end performance of the whole pipeline. We should keep in mind that the 83.66% sequence accuracy allows for zero mistakes in the predicted sequence and that this is the expected performance if we – without any pre-processing or adjustments – feed a collection of raw scans into the pipeline. Also, given the noise in the crowdsourced dataset, we can argue that 83.66% might be a slightly conservative estimate unless crowdsourcing participants and ML make the exact same mistakes on the exact same documents.

Using the ML approach, it is cheap to transcribe additional fields on the documents as it only requires a training sample. As we have seen, the training sample can be much smaller than the full collection of documents. This can be exploited to produce higher sequence accuracy rates if there are internal correspondences between the fields in the source document. For example, the death certificates contain both birth and death date and age. These three fields should be internally consistent. If they do not match then either (1) the source document contains a mistake or (2) the ML model made a mistake. If we transcribe both age and dates and exploit the correspondence between these fields, we can filter out 5,767 cases where the predicted and implied age differ by more than one year. This leaves 17,496 documents where we achieve an ML sequence accuracy of 93.56% end-to-end. Even if the

[20] Not completely identical, as we take common differences in formatting into account, so, e.g., 01-3-2000 and 1/3-2000 would be considered equal.

problematic documents need to be manually transcribed, the ML model still produces a reduction in the manual transcription burden by around 75% relative to manually transcribing the whole dataset of $23,263$ documents ($46,526$ dates). Hence, relationships between fields can provide automatic verification and be used to flag problematic records for manual review. This method can of course also be applied in a manual or crowdsourcing context, but in this case transcription of additional fields is more costly.

In addition to the accuracy rates, we also compare the data obtained from the ML and crowdsourcing approaches. Any systematic bias in the ML model would produce deviations from the empirical distribution observed in the crowdsourced dataset. Also – based on the discussion above – we expect internal consistency between ML transcribed ages and dates. Figure 8 compares kernel density estimates of the age distribution produced by (red line) ML age transcriptions, (blue line) ML date transcriptions, and (green line) crowdsourced date transcriptions. Note that the figure only displays ages in the interval $[0; 100]$, any ages outside this interval are discarded, and we discard all predicted dates where the year does not contain four digits. In some documents, the year has been abbreviated to only the last two digits (e.g., 1890 becomes 90 or 1910 becomes 10), so the leading digits could be either 18 or 19. This is not a shortcoming of the transcription model but rather a lack of information in the source documents. In Figure 8, we see that the distribution of ML age transcriptions is very close to the age distribution implied by the crowdsourced date transcriptions. The age distribution implied by the ML date transcriptions also appear similar, although with a notable difference around ages 75–85. This deviation is not surprising as the dates contain longer sequences to transcribe (relative to age) and the ML model needs to transcribe both birth and death date correctly (at least down to the year) to get an approximately correct age prediction.

Motivated by the notable difference in performance between the whole pipeline and transcription only, we manually review some of the cases where the crowdsourced and model predictions differ. This reveals, albeit qualitatively, that most discrepancies are related to

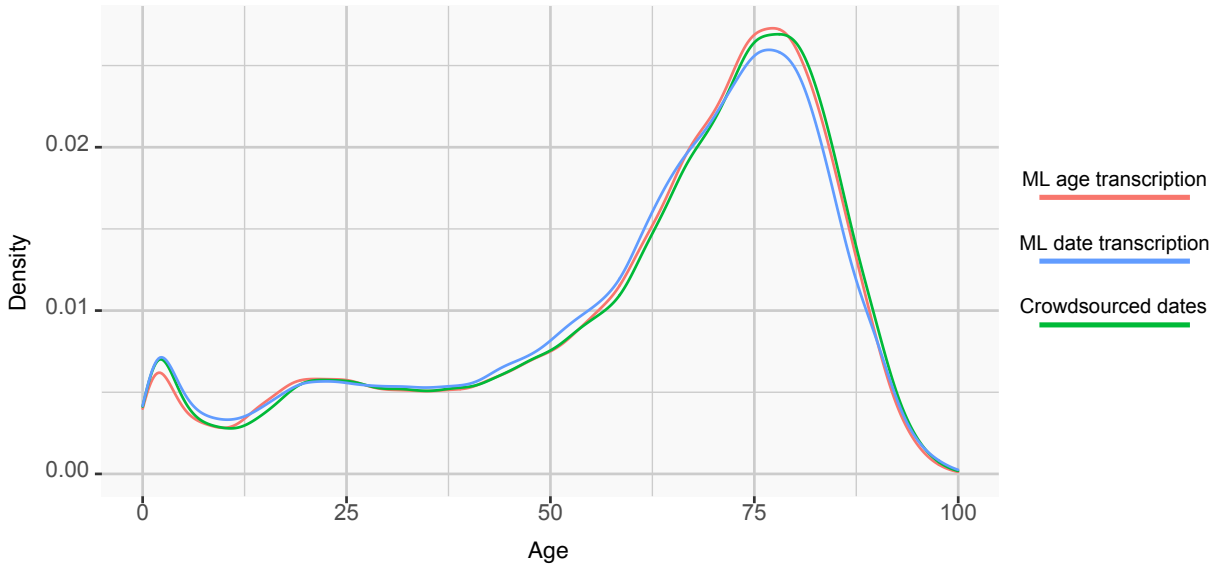

Figure 8: Lifetime distributions. Kernel density estimates of the lifetime distribution implied by ML age transcriptions (red), ML date transcriptions (blue), and crowdsourced date transcriptions (green) in an overlapping sample of $23,263$ individuals restricted to the age interval $[0; 100]$. *Implied age* refers to the difference between the birth and death dates in years.

segmentation issues where the CPD segmentation obscures the year in the date fields. The CPD segmentation template makes a cut to separate the age and death date fields. The year is the rightmost component of the date and hence most likely to be impacted by this cut, see the position of the death date and age in Figure 5. This explanation is corroborated if we look at the accuracy rates for the individual date components with the crowdsourced dataset as ground truth: Day accuracy is 95.92%, month accuracy is 96.98%, and year accuracy is 89.11%. Clearly, the accuracy for the year component is notably lower. When discovered, such issues can easily be corrected in the ML pipeline. Had the transcription been done manually, it would be much more costly to correct systematic transcription errors.

As a concluding remark, keeping in mind the resources and time needed to perform manual or crowdsourced transcription, we note that the ML end-to-end accuracy of 83.66% – trained on only 11,630 dates and 11,072 ages – might in many cases be acceptable, especially for large document collections that are otherwise infeasible to transcribe.

It should be noted that the ML model in this application has not been carefully optimised

and it has only been trained on a small training dataset. It is conceivable that the model can perform substantially better. In addition, improvements in segmentation would also positively impact the end-to-end accuracy. Also, in practical applications, the model can be used to speed up transcription while retaining manual review of each (or some) predictions. The reviewed predictions can then be used to re-train and improve the model.

There is also the possibility to rely on the confidence measures associated with model predictions and remove transcriptions where the model is highly uncertain, i.e. where the confidence measure is below a certain threshold. Subsequently, the removed transcriptions could be reviewed manually and the model could be retrained based on the additional labels.

Finally, we apply Transkribus for transcribing dates. For comparison we use the same collection of evaluation death certificates as the date model. Extraction of the date fields is difficult without a segmentation step, hence for a fair comparison of the transcription performance, we assist Transkribus by extracting the relevant table segments. The assisted Transkribus method achieves a sequence accuracy of 73.9%, compared to 90.5% using the date model and 96.3% using crowdsourcing. Not only is the performance by the assisted Transkribus significantly worse, but extracting the transcribed information by Transkribus is also more time-consuming. In terms of costs, it should also be stressed that using Transkribus to transcribe large collections may be expensive.[21]

5. DISCUSSION AND CONCLUSION

We have proposed a custom ML pipeline and we show that it can efficiently transcribe massive collections of tabular documents, hereby producing data that are directly usable in quantitative research. We test and illustrate our pipeline on two large collections of historical documents.

In the first case study, we apply the layout classification to a complex tabular dataset of

[21] The current price is €0.24 per handwritten page when buying credits in bulk for 24,000 documents at a time, see https://readcoop.eu/transkribus/credits/. For millions of documents, this is not financially feasible for small organizations or independent researchers.

nurse records and demonstrate that we can use layout analysis, 'step one' of our pipeline, in itself to collect data. We also discuss how the ML method gives additional insights on treatment assignment that complements, and expands on, the results from an intention to treat analysis, without requiring additional manual transcription of the source data.

The second case study extends this example and shows that, after the initial layout classification, we can use table segmentation and handwritten text recognition to transcribe handwritten information inside the tabular document. We also show that, based on the raw scans, the pipeline can automatically collect lifespan data that are directly useful in modeling mortality risk. In this application, we consider all three steps of the pipeline, and apply the methods to a subset of a large collection of death certificates. Finally, we also compare the ML approach to traditional crowdsourcing and Transkribus, and show that ML can significantly reduce the transcription burden for tabular documents while performing on-par with crowdsourced transcriptions and better than transcriptions from Transkribus. In particular, we find that accuracy of our transcriptions are close to those from crowdsourcing, and that the statistical distribution of the transcribed data is roughly equivalent to that of the ground truth data. This is especially important if the data are to be used in downstream statistical models.

We wish to emphasise again that our ML pipeline scales well. The cost difference between transcribing a hundred of thousand or millions of documents is negligible as opposed to the cost of the equivalent manual transcription. For our pipeline, the fixed costs of the initial setup are high, while the variable costs are very low. For off-the-shelf tools, such as Transkribus and Monk, the opposite is true. This motivates the use of a custom pipeline for large-scale projects where the variable costs dominate the initial fixed costs. Also, the initial fixed costs for a custom pipeline can be reduced by relying on the modular tools and training datasets we provide in this work.

REFERENCES

Abramitzky, Ran et al. (2021). "Intergenerational mobility of immigrants in the United States over two centuries". In: *American Economic Review* 111.2, pp. 580–608.

Acuña-Soto, Rodolfo, Joel G. Breman, et al. (2012). "The fate of historical death certificates: the silent burning of another library of Alexandria". In: *American Journal of Public Health* 102.12, E1.

Almond, D. and J. Currie (2011a). "Human capital development before age 5". In: *Handbook of Labor Economics*. Elsevier, pp. 1315–1486.

— (2011b). "Killing Me Softly: The Fetal Origins Hypothesis". In: *The Journal of Economic Perspectives* 25 (3), pp. 153–172.

Almond, Douglas, Janet Currie, and Valentina Duque (2018). "Childhood Circumstances and Adult Outcomes: Act II". In: *Journal of Economic Literature* 56.4, pp. 1360–1446. DOI: 10.1257/jel.20171164. URL: https://www.aeaweb.org/articles?id=10.1257/jel.20171164.

Angrist, Joshua D., Guido W. Imbens, and Donald B. Rubin (1996). "Identification of causal effects using instrumental variables". In: *Journal of the American Statistical Association* 91.434, pp. 444–455.

Angrist, Joshua D. and Alan B. Krueger (1991). "Does Compulsory School Attendance Affect Schooling and Earnings?" In: *The Quarterly Journal of Economics* 106.4, pp. 979–1014.

Bay, Herbert et al. (2008). "Speeded-Up Robust Features (SURF)". In: *Computer Vision and Image Understanding* 110.3, pp. 346–359.

Berg, Gerard J. van den, Maarten Lindeboom, and France Portrait (Mar. 2006). "Economic Conditions Early in Life and Individual Mortality". In: *American Economic Review* 96.1, pp. 290–302. DOI: 10.1257/000282806776157740. URL: https://www.aeaweb.org/articles?id=10.1257/000282806776157740.

Besl, Paul J. and Neil D. McKay (1992). "Method for registration of 3-D shapes". In: *Sensor Fusion IV: Control Paradigms and Data Structures*. Vol. 1611. International Society for Optics and Photonics, pp. 586–606.

Biering-Sørensen, Fin, Jørgen Hilden, and Knud Biering-Sørensen (1980). "Breast-feeding in Copenhagen, 1938-1977: Data on more than 365,000 infants". In: *Danish Medical Bulletin* 27, pp. 42–48.

Bjerregaard, Lise G. et al. (2014). "Effects of body size and change in body size from infancy through childhood on body mass index in adulthood". In: *International Journal of Obesity* 38.10, pp. 1305–1311.

Bluche, Théodore, Jérôome Louradour, and Ronaldo Messina (2017). "Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE, pp. 1050–1055.

Bruckner, Tim A. et al. (2014). "Ambient temperature during gestation and cold-related adult mortality in a Swedish cohort, 1915–2002". In: *Social Science & Medicine* 119, pp. 191–197. ISSN: 0277-9536. DOI: https://doi.org/10.1016/j.socscimed.2014.01.049. URL: https://www.sciencedirect.com/science/article/pii/S0277953614000768.

Chen, Nawei and Dorothea Blostein (2007). "A survey of document image classification: problem statement, classifier architecture and performance evaluation". In: *International Journal of Document Analysis and Recognition (IJDAR)* 10.1, pp. 1–16.

Cho, Kyunghyun et al. (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 1724–1734.

Clinchant, S. et al. (2018). "Comparing Machine Learning Approaches for Table Recognition in Historical Register Books". In: *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pp. 133–138.

Cohen, Joachim et al. (2007). "Using death certificate data to study place of death in 9 European countries: opportunities and weaknesses". In: *BMC public health* 7.1, pp. 1–9.

Colavizza, Giovanni, Maud Ehrmann, and Fabio Bortoluzzi (2019). "Index-Driven Digitization and Indexation of Historical Archives". In: *Frontiers in Digital Humanities* 6, p. 4. ISSN: 2297-2668. DOI: 10.3389/fdigh.2019.00004. URL: https://www.frontiersin.org/article/10.3389/fdigh.2019.00004.

Coüasnon, Bertrand and Aurélie Lemaitre (2014). "Recognition of Tables and Forms". In: *Handbook of Document Image Processing and Recognition*. Ed. by David Doermann and Karl Tombre. London: Springer London, pp. 647–677.

Csurka, Gabriella et al. (2004). "Visual Categorization with Bags of Keypoints". In: *Workshop on Statistical Learning in Computer Vision, ECCV*. Vol. 1. 1-22. Prague, pp. 1–2.

Ester, Martin et al. (1996). "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *KDD-96 Proceedings*. AAAI Press, pp. 226–231.

Feigenbaum, James J (2018). "Multiple measures of historical intergenerational mobility: Iowa 1915 to 1940". In: *The Economic Journal* 128.612, F446–F481.

Gill, James R. and Maura E. DeJoseph (July 2020). "The Importance of Proper Death Certification During the COVID-19 Pandemic". In: *JAMA* 324.1, pp. 27–28. ISSN: 0098-7484. DOI: 10.1001/jama.2020.9536. URL: https://doi.org/10.1001/jama.2020.9536.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. MIT Press.

Graves, Alex, Santiago Fernández, et al. (2006). "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks". In: *Proceedings of the 23rd International Conference on Machine learning*, pp. 369–376.

Graves, Alex, Marcus Liwicki, et al. (2008). "A novel connectionist system for unconstrained handwriting recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 31.5, pp. 855–868.

Gupta, Ankush, Andrea Vedaldi, and Andrew Zisserman (2018). "Learning to Read by Spelling: Towards Unsupervised Text Recognition". In: *Proceedings of the 11th Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 1–10.

Gutmann, Myron P., Emily Klancher Merchant, and Evan Roberts (2018). ""Big Data" in Economic History". In: *The Journal of Economic History* 78.1, pp. 268–299.

Harris, Christopher and Mike Stephens (1988). "A combined corner and edge detector". In: *Proceedings of the Alvey Vision Conference*, pp. 147–151.

He, K., G. Gkioxari, et al. (2017). "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988.

He, Kaiming, Xiangyu Zhang, et al. (2016). "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hoehn-Velasco, Lauren (2021). "The Long-term Impact of Preventative Public Health Programs". In: *The Economic Journal* 131.634, pp. 797–826.

Högberg, Ulf and Stig Wall (1986). "Secular trends in maternal mortality in Sweden from 1750 to 1980". In: *Bulletin of the World Health Organization* 64.1, p. 79.

Imbens, Guido W. and Donald B. Rubin (Oct. 1997). "Estimating Outcome Distributions for Compliers in Instrumental Variables Models". In: *The Review of Economic Studies* 64.4, pp. 555–574. ISSN: 0034-6527. DOI: 10.2307/2971731. URL: https://doi.org/10.2307/2971731.

James, Gareth et al. (2013). *An Introduction to Statistical Learning*. New York: Springer.

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization". In: *arXiv preprint 1412.6980*. URL: https://arxiv.org/abs/1412.6980.

Lee, Chen-Yu and Simon Osindero (2016). "Recursive Recurrent Nets With Attention Modeling for OCR in the Wild". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2231–2239.

Li, X., L. Wang, and Y. Fang (2019). "PC-Net: Unsupervised Point Correspondence Learning with Neural Networks". In: *2019 International Conference on 3D Vision (3DV)*, pp. 145–154.

Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2, pp. 91–110.

Moss, Michael, David Thomas, and Tim Gollins (2018). "The Reconfiguration of the Archive as Data to Be Mined". In: *Archivaria*, pp. 118–151.

Muehlberger, G. et al. (2019). "Transforming scholarship in the archives through handwritten text recognition: Transkribus as a case study". In: *Journal of Documentation* 75, pp. 954–976. URL: https://doi.org/10.1108/JD-07-2018-0114.

Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*. Cambridge, Massachusetts: MIT Press.

Myronenko, Andriy and Xubo Song (2010). "Point Set Registration: Coherent Point Drift". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.12, pp. 2262–2275.

Nion, T. et al. (2013). "Handwritten Information Extraction from Historical Census Documents". In: *12th International Conference on Document Analysis and Recognition*, pp. 822–826.

Pan, Sinno Jialin and Qiang Yang (2009). "A Survey on Transfer Learning". In: *IEEE Transactions on Knowledge and Data Engineering* 22.10, pp. 1345–1359.

Rosenzweig, Roy (June 2003). "Scarcity or Abundance? Preserving the Past in a Digital Era". In: *The American Historical Review* 108.3, pp. 735–762. ISSN: 0002-8762. DOI: 10.1086/ahr/108.3.735. URL: https://doi.org/10.1086/ahr/108.3.735.

Schomaker, Lambert (2020). "Lifelong Learning for Text Retrieval and Recognition in Historical Handwritten Document Collections". In: *Handwritten Historical Document Analysis, Recognition, and Retrieval – State of the Art and Future Trends*. Ed. by Andreas

Fischer, Marcus Liwicki, and Rolf Jurg Ingold. Chap. Chapter 12, pp. 221–248. DOI: 10.1142/9789811203244_0012. URL: https://www.worldscientific.com/doi/abs/10.1142/9789811203244_0012.

Simonsen, Lone et al. (2011). "The need for interdisciplinary studies of historic pandemics". In: *Vaccine* 29.Supplement 2.

Simonyan, Karen and Andrew Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations"*. URL: https://arxiv.org/abs/1409.1556.

Sivic, Josef and Andrew Zisserman (2009). "Efficient Visual Search of Videos Cast as Text Retrieval". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.4, pp. 591–606.

Szeliski, Richard (2010). *Computer Vision: Algorithms and Applications*. New York: Springer.

van der Maaten, Laurens and Geoffrey Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.Nov, pp. 2579–2605.

Weber, Andreas et al. (2018). "Towards a Digital Infrastructure for Illustrated Handwritten Archives". In: *Digital Cultural Heritage: Final Conference of the Marie Skłodowska-Curie Initial Training Network for Digital Cultural Heritage, ITN-DCH 2017, Olimje, Slovenia, May 23–25, 2017, Revised Selected Papers*. Ed. by Marinos Ioannides. Cham: Springer International Publishing, pp. 155–166. ISBN: 978-3-319-75826-8. DOI: 10.1007/978-3-319-75826-8_13. URL: https://doi.org/10.1007/978-3-319-75826-8_13.

Xu, Kelvin et al. (2015). "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: *International Conference on Machine Learning*, pp. 2048–2057.

Ye, Junting and Steven Skiena (2019). "The Secret Lives of Names? Name Embeddings from Social Media". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3000–3008.

A. TECHNICAL APPENDIX

A.1. Methods – Nurse records

The documents are scanned and stored digitally as image files. Images consist of coloured dots called pixels. Each pixel is characterised by a location and a colour. Stacking a certain number of pixels horizontally and vertically forms an image. Thus, we can consider an image of $h \times w$ pixels to be an $h \times w$ matrix where each entry corresponds to a single pixel. In grayscale photos, each pixel can only attain white, black, and shades in-between. This is represented by a byte (8 bits) specifying a value between 0 and 255 with 0 being black and 255 white. The core of the machine digitisation process can be formulated as various statistical learning problems where we model different aspects of the visual information to learn a mapping from the image matrix into a representation that is suitable for analysis. Learning this mapping represents an array of challenges. In particular, this is often challenging because the image matrix can be of very high dimension. A feature is a lower-dimensional variable that captures some aspect of the high-dimensional image, and hopefully in a way that is more informative than the raw image data itself. Convolutional neural networks, see Goodfellow et al. (2016, Chp. 9), learn to extract such features when trained for image classification and it turns out that these features are generally informative despite being trained on a specific dataset (Simonyan and Zisserman, 2015). This property is exploited in transfer learning where parts of a neural network trained on one set of images are applied for a new task on a different set of images (Pan and Yang, 2009). The VGG16 network is an example of a deep convolutional neural network that was trained on over 1 million photos to distinguish between $1,000$ objects (Simonyan and Zisserman, 2015). Based on the concept of transfer learning, we use this pre-trained network to extract features from the journal images. This is a useful trick that can provide informative features without the need to train more sophisticated feature extractors or models. It works similar to traditional feature extractors, e.g., SIFT (Lowe, 2004) and SURF (Bay et al., 2008), but the feature representation is learned instead

Ground truth	ML detection: Treated	ML detection: Not Treated	
Treated	234	0	234
Not Treated	0	3,766	3,766
	234	3,766	

Table A.1: Confusion matrix for the ML treatment detection model. The frequencies are based on a randomly sampled and manually reviewed validation set of 4,000 journals (10,914 pages). The treatment detection model does not rely on any segmentation, but detects the presence of the whole page containing the treatment table.

of manually engineered. The classification part of VGG16 is discarded – we do not care about the original classification task – and we only keep the convolutional part. Each journal page is passed through the VGG16 convolutional network and we obtain a 512-dimensional feature vector for each page that describes some aspects of the visual information.

We use unsupervised methods to explore the features. The features are clustered using DBSCAN (Ester et al., 1996) – a density-based clustering algorithm. Pages with similar layout should cluster together as they share a similar VGG16 feature vector. To visualise the features and clusters, we embed them in a two-dimensional space with t-distributed Stochastic Neighbour Embedding (t-SNE) (van der Maaten and Hinton, 2008). The t-SNE method provides a convenient way to visualise high-dimensional spaces in two or three dimensions while retaining the local structure between points, i.e. points that are close in the high-dimensional space also tend to be close in the low-dimensional embedding space. The t-SNE embeddings for the nurse records are depicted in Figure 4. Each point represents a page in a journal and the points are colored according to their assigned cluster. A clear structure is evident. There are 37 clusters which we manually review and annotate according to their contents. Annotation is carried out by randomly sampling 10 pages from each cluster and assigning a label for the whole cluster based on the contents of these pages. This amounts to 370 journal pages that need manual review (out of 261,926). This procedure shows that the treatment pages are contained in four distinct clusters. We extract all pages residing in these clusters and classify the underlying individuals as treated, i.e. an individual is treated

if any page in their journal belongs to one of the four treatment page clusters.

A.2. Methods – Death certificates

A.2.1. Training and evaluation data

We use various datasets to train and evaluate the performance of different parts of our ML pipeline for the death certificates. Table A.2 provides an overview of the sizes of the different training and evaluation datasets, and we provide additional details on the individual datasets below.

(1) **Layouts**: The training and evaluation datasets contain 7,000 and 2,184 pages, respectively. They are used to train and evaluate the layout classification model for detecting certificates of type B. The images are of varying size and the ground truth layout type is stored as an indicator variable. The images are similar to those shown in Figure A.3.

(2) **Dates**: The training and evaluation datasets contain 11,630 and 1,000 pages, respectively. Data is approximately balanced between birth and death dates. The datasets are used to train and evaluate the transcription model for birth and death dates. Images are 320×50 pixels and the ground truth transcriptions are stored as strings in a standardised format. See Figure A.1 for examples.

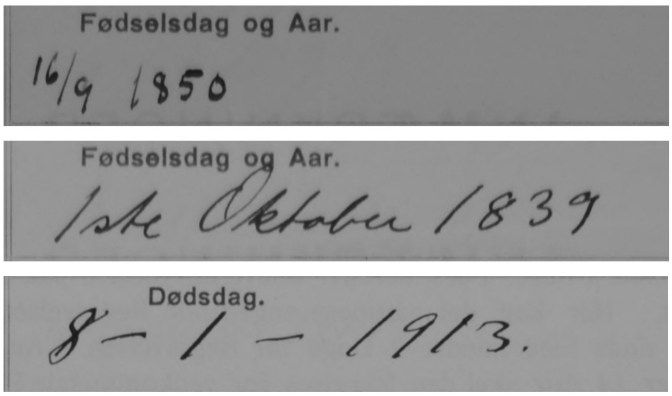

Figure A.1: Examples of the images in the date datasets.

(3) **Ages**: The training and evaluation datasets contain 11,072 and 1,000 ages, respectively. The datasets are used to train and evaluate the transcription model for age.

	Training	Evaluation	Image size
Layouts	7,000	2,184	Variable
Dates	11,630	1,000	320×50 pixels
Ages	11,072	1,000	230×75 pixels
Crowdsourced dates	-	46,526	320×50 pixels

Table A.2: Overview of the number of samples in each of the training and evaluation datasets used in the ML pipeline

Images are 230×75 pixels and the ground truth transcriptions are stored as strings and exclude the age suffix, i.e., years, months, days, or hours. Non-integer ages are also excluded. See Figure A.2 for examples.

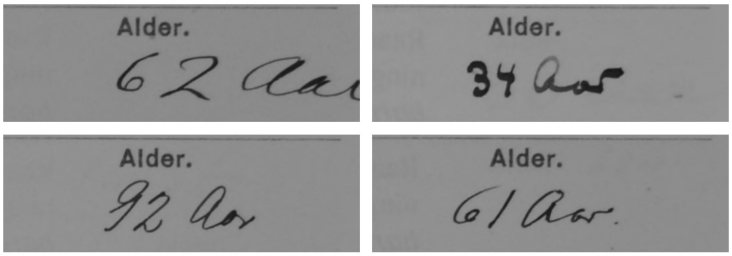

Figure A.2: Examples of the images in the age datasets.

(4) **Crowdsourced dates**: A dataset containing 23,263 complete death certificates that intersect with our collection of death certificates. Transcriptions are only available for birth and death dates, not for age. This dataset is used for evaluating the end-to-end performance of the pipeline.

A.2.2. Layout Classification

Layout classification refers to the process of organising a collection of documents by common layout structure, e.g., a common table, heading, or pre-printed landmarks. Chen and Blostein (2007) provide a relevant but methodologically outdated introduction to this problem. The layout type is important as the table segmentation step relies on a pre-defined template that must match with the pre-printed structure in the image. Due to variations in layout across

the collection of death certificates (see Figure A.3), we need to construct one template for each type. Every time we fit a template to a certificate, we need the template and certificate types to match. This is straightforward if the certificates are sorted according to layout, as in this case, all images in a given class will share the same template.

As we saw in Figure A.3, an image $\mathbf{X}_i$ of a death certificate can belong to one of K layout types that can be distinguished visually, e.g., "Type B", "Type A", and so on. Let the layout type of image i be $Y_i = k$ with $k = 1, 2, ..., K$. The K types do not need to mimic the visual types in the documents: We can easily focus on "Type B" and label everything else as "Other". We are interested in learning a model for the probability $\mathbb{P}(Y_i = k \,|\, \mathbf{X}_i)$ such that we can infer the most likely layout type $\hat{y}_i = \arg\max_k \mathbb{P}(Y_i = k \,|\, \mathbf{X}_i)$. This resembles a conventional K-class classification problem with the only difference that $\mathbf{X}_i$ is an image.

We already saw an example of unsupervised layout classification in the context of the nurse journals in Section A.1 where we emphasised the importance of constructing a lower-dimensional feature that describes the raw image. In the supervised setting, there are two main considerations: (1) what features should we use, i.e. how do we construct a feature $g(\mathbf{X}_i)$ that represents the high-dimensional information in $\mathbf{X}_i$, and (2) what classifier should we use, i.e. what model do we choose for the probability $\mathbb{P}(Y_i = k \,|\, g(\mathbf{X}_i))$. This cannot be answered definitively and depends on the application. There are many equally viable approaches. In the nurse journal application, we solved (1) by using the features from a pre-trained neural network (VGG16) and used these directly in a cluster analysis. A modern supervised end-to-end approach would be to train a convolutional neural network (CNN) to classify the pages using raw images as input, i.e. $g(\mathbf{X}_i) = \mathbf{X}_i$. In this case, the neural network will solve both (1) and (2) as it learns both the feature extractor and the classifier during training.

We take a simpler approach and use the visual Bag-of-Words (BoW) method to classify the layout type of the certificates. This method provides an intuitive definition of features in terms of visual landmarks or "words". BoW is a technique originally developed to classify

Figure A.3: A sample of the different pages in the collection of death certificates. The examples are not exhaustive.

chunks of text according to the content (Murphy, 2012, p. 87). It operates by determining the frequency of words (i.e. the frequency of some global set of words – the bag-of-words) in each text document. The method has been applied successfully in the field of computer vision, see Csurka et al. (2004) and Sivic and Zisserman (2009), where we instead operate with a bag of visual words, i.e. chunks of images.

The visual bag-of-words model creates features on the basis of a codebook or dictionary. This dictionary is constructed by extracting key points from a training dataset and clustering them such that we obtain M groups of key points where key points within a group appear similar in some sense. If we think of a training dataset that consists of photographs of animals, we could have a key point cluster related to eyes, one for ears, etc. Each of these clusters is a *visual word*. The visual words are similar to the feature clusters we discovered in the nurse journals using the unsupervised method, i.e. those in Figure 4. Here we extract the features using Speeded-Up Robust Features (SURF) (Bay et al., 2008) as this has historically been the common choice, see Csurka et al. (2004).

When the dictionary has been constructed, we are ready to create the actual feature vectors for the images. For a given training image i, we extract SURF key points and assign each of these to the M key point clusters based on distance. We then count the number of features from the image that belongs to each of the M key point clusters and construct a vector of normalised frequencies which serves as the final feature vector of the image. Note that the size M of the dictionary determines the dimensionality of the final feature vector. In the classification step, we train a model to classify each feature vector (and thus each image) into one of the K classes. The classifier can be of any type. Here we use support vector machines with radial basis kernels, see the introduction in James et al. (2013, Chp. 9). There are several tuning parameters in the BoW model (dictionary size, margin, etc.) which we selected using leave-one-out cross-validation. The computational burden lies in the initial keypoint extraction and clustering. When the appropriate features have been extracted, it is very fast to fit the actual classifiers and hence it is computationally fast to

do cross-validation for the classifier hyperparameters.[22] For an end-to-end neural network it is substantially slower to do hyperparameter optimisation as the whole network will need to be re-trained – possibly for hours – for each set of hyperparameters. This highlights that simpler classifiers can be useful as a first step before developing more complex models.

A.2.3. Table Segmentation

In the table segmentation step, the goal is to extract smaller images corresponding to each field (or cell) of a larger form (or table) in the source image. Coüasnon and Lemaitre (2014) provide a general introduction to the topic. There are two components to this problem: (1) identifying the structure of the table in the source image and (2) exploiting the structure to extract the field images. We apply a simple template-based approach where a predefined template is fitted over the source image using a set of landmark points.

In ongoing work, we are considering how to solve (1) using edge-detection neural networks. However, in this paper we rely on standard filtering operations from the computer vision literature to binarise the source image and find straight horizontal and vertical lines. This implies that we are not using ML and since the operations are application dependent we only briefly discuss this part in the results section. We solve (2) using point set registration methods where we apply the Coherent Point Drift (CPD) method of Myronenko and Song (2010). This method relies on a probabilistic model where we learn a transformation between two sets of points using maximum likelihood and use it to directly fit a template to the source image.

A point set is – as the name implies – simply a set of points. Since we are working with images, these points reside in the plane and are characterised by two coordinates. Let $P_1 = \{(x_{11}, y_{11}), (x_{12}, y_{12}), ..., (x_{1N}, y_{1N})\}$ be a set of points that define a template (e.g. the corners of a table) and analogously let $P_2 = \{(x_{21}, y_{21}), (x_{22}, y_{22}), ..., (x_{2K}, y_{2K})\}$ be a set of points in an image that we think roughly corresponds to those in the template. Note that

[22] Note that for large training sets, the polynomial runtime for support vector machines is a limitation and another classifier should be used.

the number of points in the two sets do not need to be the same. Point registration is the problem of aligning the points in P_1 (the template) over the points in P_2 (the image) (Besl and McKay, 1992). This is illustrated in Figure A.5, where we align template points P_1 (blue dots) with the image landmarks P_2 (red dots). We can intuitively understand this as finding a transformation function T that applies some transformation to all points in P_1 such that $T(P_1)$ aligns with P_2 in some distance metric d. The point set registration methods differ in their choice of distance d and the constraint they put on the transformation T. The non-rigid version of the CPD method in Myronenko and Song (2010) assumes that the points in the image P_2 are generated by a Gaussian distribution "around" the template points P_1 and it puts only mild regularity conditions on the transform T. In particular, it assumes that points move freely but coherently. Myronenko and Song (2010) model the problem using a Gaussian mixture model. We rely directly on their algorithm to align our template to the image by supplying our template points P_1 and image landmarks P_2. The algorithm applies the standard Expectation Maximisation (EM) method to solve the maximum likelihood problem.

It has been suggested that the CPD method can be improved by using a neural network to learn the transformation (Li et al., 2019) but we do not consider this here. There are also examples of more general learning based table segmentation that does not rely on a pre-specified template, see, e.g., Clinchant et al. (2018).

The initial template is constructed by finding a well-scanned death certificate where we can manually extract a number of points which can be used as anchors for the template, see Figure 2. This is only done once and these points comprise the template point set P_1. After the template has been established, each type B death certificate is subjected to a set of morphological operations (erosion and dilation) to find pixels belonging to table rows or columns. The image is first thresholded (i.e. converted from gray scale into black/white where all pixels are either 0 or 255) and afterwards erosion and dilation operations are applied. Erosion and dilation are common non-probabilistic ways to extract straight (vertical

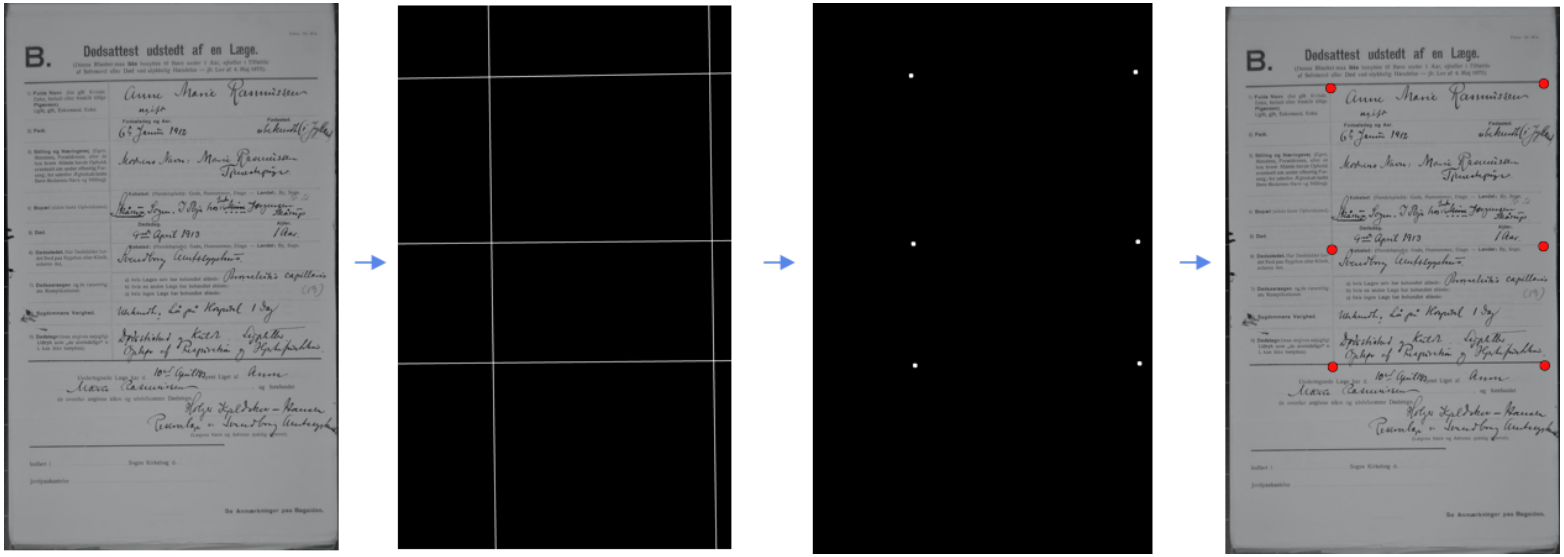

Figure A.4: Landmark detection using morphological operations and a corner detector.

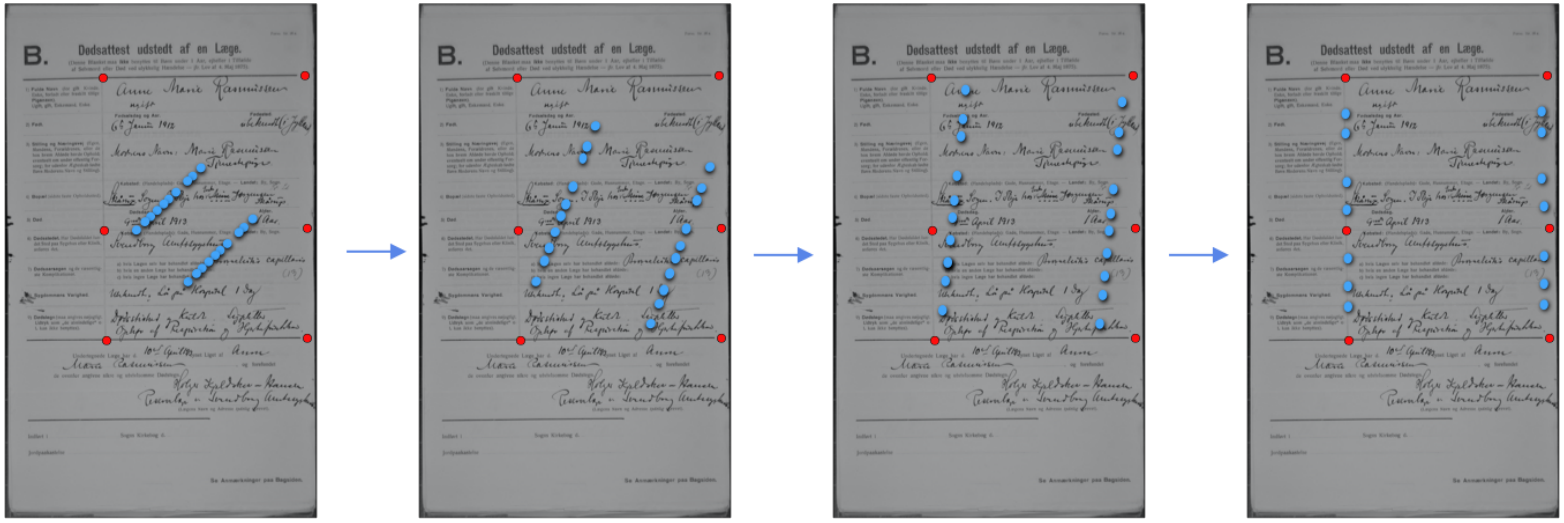

Figure A.5: Point set alignment. Blue dots represent the template while red dots are the landmarks found in the source document.

or horizontal) lines in computer vision problems, see, e.g., Szeliski (2010). The morphological operations require careful hand-tuning to find the optimal parameters for the document type but these parameters only need to be tuned once for the whole collection. When the image has been reduced to straight white lines on a black background, we detect the intersection points of the lines using the Harris corner detection algorithm (Harris and Stephens, 1988). The detected corners provide a number of landmark points that inform us on the location, scale, rotation, etc. of the table. The first two images in Figure A.4 display a death certificate before and after application of the morphological operations, the third image shows the corners detected by the Harris algorithm, and the fourth image shows the final identified landmark points. The coordinates of the landmarks comprise the point set P_2. Next, we apply the CPD algorithm to align the template points in P_1 to the image landmarks in P_2. This iterative fitting process is illustrated in Figure A.5. Even if extra landmark points are detected, the CPD algorithm remains robust as long as the detected landmark points P_2 are sufficiently spread out along the axes, i.e. if the noise is fairly uniform. Once the points have been aligned, we use the estimated transformation T to fit the template onto the image. The template then guides the cuts that separate the larger image into individual field images. This is illustrated in Figure A.6. Figures 6 and 7 show examples of the final segmented fields for dates. We do not report quantitative measures of segmentation performance but note that the end-to-end transcription accuracy in Section 4 will include any errors produced by the segmentation step if these make the field images unreadable.

The field images could be further segmented to limit noise and assure that the position and scale of the text is similar between samples. A possible way to implement this is a Mask R-CNN network (He, Gkioxari, et al., 2017) which could be trained to predict regions of noise and handwritten text respectively. The text regions are then extracted and re-aligned on a blank image. However, this adds further complexity to the problem by introducing an additional segmentation model. The Mask R-CNN would need training data which requires manual annotation of text outlines to create a ground truth dataset. We

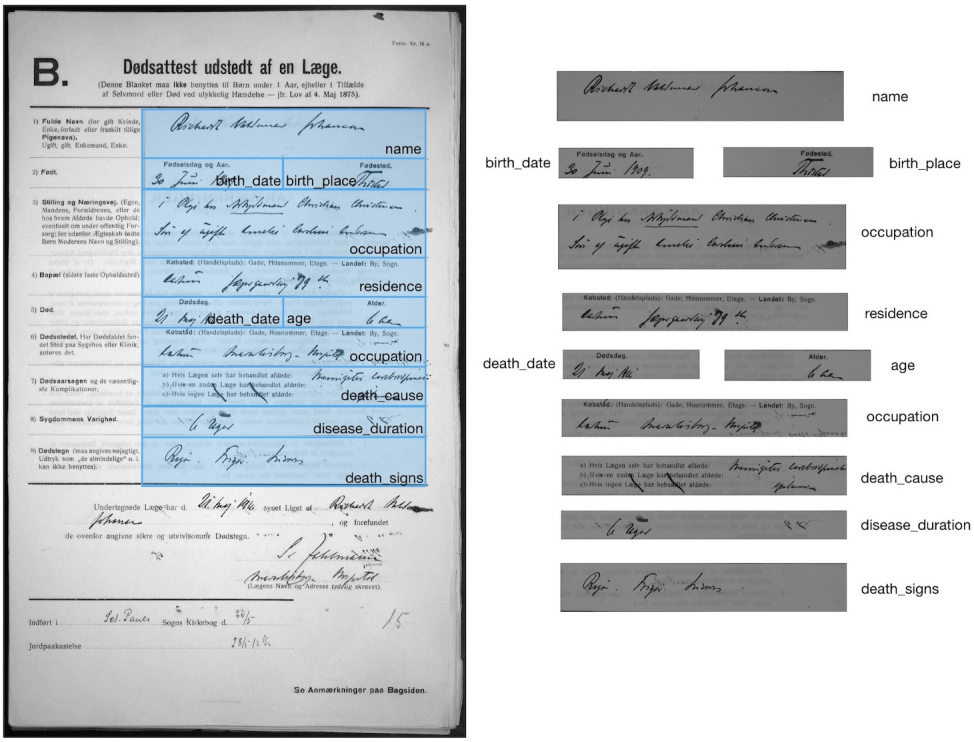

Figure A.6: Example of field images extracted using the fitted template.

will instead run the transcription model directly on the field images without any additional segmentation/cleaning and show that this is a viable approach.

A.2.4. Cell transcription

Transcription is the process of converting an image of text into a string representation. Assume we have sequences of the form $Y_i = (Y_{i,1}, Y_{i,2}, ..., Y_{i,T})$ where T is the maximum sequence length and each $Y_{i,t}$ is a random variable on the sample space

$$\Omega = \{\texttt{<Token1>}, \texttt{<Token2>}, ..., \texttt{<TokenN>}\}.$$

We call Ω a dictionary (or token space) and the tokens $\{\texttt{<Token1>}, ..., \texttt{<TokenN>}\}$ serve as placeholders that can contain any information, be it individual characters, words, numbers, special symbols, or any combination thereof.[23] The contents of the dictionary depend on

[23]There are also sequence models relying on vector-space embeddings of words, such as the image captioning model of Xu et al. (2015). The output from the learned mapping will then be a vector in $\mathbb{R}^K$ with

the application and there are essentially no restrictions. For example, if Ω contains all characters in the alphabet then we can represent any conventional word of length T as a sequence $Y_i \in \Omega^T$. Similarly, if Ω contains whole words then we can represent a sentence of T words equivalently as $Y_i \in \Omega^T$. Based on an image $\mathbf{X}_i$ we want to predict a sequence $Y_i = f(\mathbf{X}_i)$, where f is some unknown map from images to sequences. Learning the sequence mapping f is the primary transcription problem.

The optical character recognition (OCR), handwritten text recognition (HTR), and scene-text detection literature are all concerned with this problem, but the solutions differ heavily in how they model f and how they choose their tokens, i.e. how they choose Ω. Some methods rely on character-by-character transcription where Ω is simply the alphabet (Graves, Liwicki, et al., 2008) and hence these models are, in theory, not limited in the set of words they can transcribe. Other approaches, like vector-space models, rely on (embeddings of) whole words such as the image captioning model in Xu et al. (2015). In some cases, these choices limit the generality of the method, i.e. the whole-word models will not be able to transcribe words that are not in the dictionary. In our case, we are mainly interested in (1) numerical information such as dates, numbers, and area codes, and (2) string information such as names, locations and, causes-of-death. All of these fields have dictionaries that we can easily construct. In the contemporary literature, it is common to use neural networks to model f with various degrees of sophistication, see, e.g., the introduction in Graves, Liwicki, et al. (2008). The majority of the work has been on supervised methods where models are trained on pairs $(\mathbf{X}_i, Y_i)$ but there are examples of unsupervised approaches as well, such as Gupta et al. (2018).

We limit our attention to supervised models and focus on situations where the dictionary is small, well-defined, and consists of a few words and characters, i.e. a *constrained*

K being the embedding space dimension. This vector is then mapped to the closest (in some metric) known word in this space by a deterministic function, hence the final prediction will still be of the form $Y_i \in \Omega^T$. However, word embeddings are usually constructed by performing an auxiliary task on a large dataset of text, and it is not entirely obvious how such an approach would work for other things like cause-of-death or name, i.e. what properties would we expect from reasonable "name" embeddings? See also Ye and Skiena (2019) who use twitter data to create an embedding space for names.

transcription model. Our intuition is that there is no reason to allow predictions such as "Bob" and "Apples" if we know that these should never occur in the data. For example, to represent any standard date we only need the digits $0-9$, names of the months, and some special characters.[24]

Model. We utilise an attention-based neural network suggested by Xu et al. (2015) for image captioning and repurpose it for transcription of handwritten text. Originally, the model by Xu et al. (2015) predicts a string of words that describes the contents of an input image, e.g., if it is supplied with an image of a bird sitting in a tree then the model would predict the sentence "A bird sitting in a tree". However, the model of Xu et al. (2015) is generally applicable to tasks that involve image inputs and sequence outputs – which is exactly what characterises the transcription problem. A similar attention-based model for OCR was proposed by Lee and Osindero (2016) and it has been demonstrated that more complex multi-head attention models can transcribe whole paragraphs of handwritten text (multiple lines) without prior line segmentation, see Bluche et al. (2017).

In our implementation of Xu et al. (2015), we replace the final embedding output with a softmax layer. This allows the model to predict probabilities for each of the tokens in our dictionary Ω. In particular, assume that we have an image $\mathbf{X}_i$ and we are at step t in the prediction of the corresponding sequence $Y_i = (Y_{i,1}, ..., Y_{i,T})$. The model would predict the step t conditional probabilities

$$\mathbb{P}(Y_{i,t} = k \,|\, \mathbf{X}_i, Y_{i,t-1} = \hat{y}_{i,t-1}, ..., Y_{i,1} = \hat{y}_{i,1}), \quad k \in \Omega$$

where $(\hat{y}_{i,t-1}, ..., \hat{y}_{i,1})$ are the previously predicted tokens in the sequence.

The model architecture is depicted in Figure A.7. First, the feature network encodes the input image into a feature map. The attention mechanism A then applies soft-attention to

[24] This is less straightforward for transcription of names. However, it is possible to construct dictionaries based on the most common names and only transcribe these. Our initial tests of this approach have been promising for reading names in heavily cluttered and poorly segmented fields, but we do not present the results here.

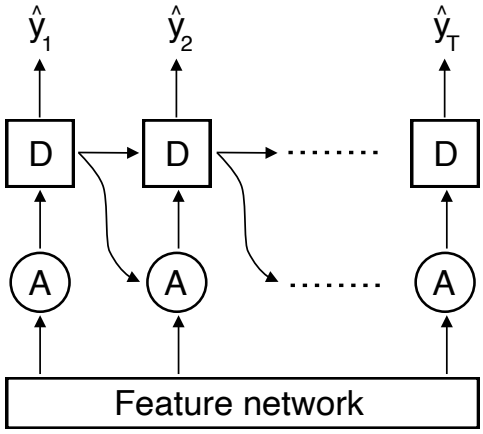

Figure A.7: Transcription model based on the image captioning model in Xu et al. (2015). The feature network extracts features from the input image. The attention mechanism, denoted by A, weighs the features and supply these as input to an RNN cell, D, that predicts one token of the sequence. The hidden state of the RNN cell is updated, and the updated state is passed to the attention network to decide on the next weights for the feature map. The process repeats until an end-of-sequence prediction is made.

this feature map by predicting weights in $[0, 1]$ and taking the element-wise product of the features and weights. As noted by Xu et al. (2015), soft-attention is fully differentiable so we can use standard back-propagation to train the network.[25] The attention-weighted feature map is passed to a recurrent neural network (RNN) cell D. This cell predicts the next token in the sequence and updates its hidden state. The hidden state is fed to the attention mechanism which updates the attention weights, i.e. "where should we look next"? This supplies the recurrent cell with a new weighting of the feature map. The RNN cell uses the hidden state and the weighted features to predict the next token in the sequence. This procedure is repeated until an end-of-sequence token is predicted by the model.

An alternative approach is a neural network with the Connectionist Temporal Classification (CTC) loss. Such networks have achieved state-of-the-art results on unconstrained transcription (Graves, Liwicki, et al., 2008), i.e. in settings where there are no restrictions on the words the model can transcribe – implying that Ω contains the alphabet. Graves, Fernández, et al. (2006) suggested the CTC loss for sequence labelling. The CTC-based

[25] In our work, we have found that the soft attention mechanisms are significantly easier to train compared to their hard attention counterparts. This is also noted by Lee and Osindero (2016).

networks slice the entire input image into a sequence of feature vectors and for each feature vector make a prediction of the character/token that this particular slice contains. These predictions are then combined with the CTC loss function to handle duplicate predictions and ensure alignment of the predicted sequence with the ground truth during training. Usually, the CTC networks require more careful pre-processing and line segmentation. For example, Graves, Liwicki, et al. (2008) employ substantial line segmentation and cleaning before running the images through the transcription model. It is doubtful that these models would work well on the raw field images from the death certificates without further processing. We want to limit the amount of pre-processing as much as possible to simplify the pipeline and hence have not considered the CTC approach here. However, in ongoing work we are looking to apply CTC for token-based transcription and compare the attention-based model against the CTC-based US census transcription model being developed at the BYU Record Linking Lab.[26]

Prediction. Given an image $\mathbf{X}_i$ we want to predict the corresponding sequence $Y_i \in \Omega^T$. At each sequence step t, we use the model to predict probabilities over the tokens in the dictionary Ω. Due to dependence in the sequence, the predicted probabilities across the dictionary at step t depend on all the previously predicted tokens at steps $(t-1, t-2, ..., 1)$. Exhaustive search over all possible combinations of tokens quickly becomes infeasible. As an example, the dictionary of the date transcription model contains 25 possible tokens, and with 11 elements in the sequence, we would have to enumerate and score 25^{11} different sequences for each sample we want to predict. Hence it is necessary to rely on greedy algorithms to find a feasible (but possibly suboptimal) solution. We use beam search as suggested by Xu et al. (2015).

Evaluation. We evaluate the performance of the transcription models using two metrics. One relates to the average accuracy across individual tokens and the other to the accuracy of complete sequences. To fix notation let $y_i = (y_{i,1}, ..., y_{i,k_i})$ be a realised ground truth sequence

[26] See https://rll.byu.edu/

and $\hat{y}_i = (\hat{y}_{i,1}, ..., \hat{y}_{i,h_i})$ a predicted sequence for image $\mathbf{X}_i = \mathbf{x}_i, i = 1, ..., n$ where n is the size of the evaluation dataset. Assume that the ground truth sequence is always padded so $k_i \geq h_i$ and that the padding value is chosen such that $y_{i,j} \neq \hat{y}_{i,j}$ for $j > k_i$. Moreover, let $I(\cdot)$ be the indicator function with some predicate. We define the Token Accuracy (TA) and m-error Sequence Accuracy (SA_m) by

$$\text{TA} = \frac{1}{\sum_i k_i} \sum_{i=1}^{n} \sum_{j=1}^{k_i} I(x_{i,j} = \hat{x}_{i,j}), \tag{A.1}$$

$$\text{SA}_m = \frac{1}{n} \sum_{i=1}^{n} I\left(\sum_{j=1}^{k_i} I(x_{i,j} \neq x_{i,j}) \ \leq \ m \right). \tag{A.2}$$

TA has an interpretation as the average accuracy across all predicted tokens. SA_m is the proportion of correctly predicted sequences when m mistakes are allowed in the sequence. The token and sequence accuracies are closely related to the character and word accuracies in the HTR literature, see, e.g., Graves, Liwicki, et al. (2008). However, our dictionary contains tokens which can be chosen arbitrarily, i.e. a token can be a character or a word.

Note that both TA and SA are sensitive to alignment, so while a single substitution produces one sequence error, missing or adding an extra token would misalign the entire sentence. For example, in the following prediction the model misses a single token but this causes four errors in the predicted sequence because of misalignment:

Ground truth:	`<Start>`	`<1>`	`<2>`	`<7>`	`<0>`	`<End>`
Prediction:	`<Start>`	`<1>`	`<7>`	`<0>`	`<End>`	`<Pad>`
	Correct	Correct	Error	Error	Error	Error

Hence missing a token can impose a heavy accuracy penalty when using these measures. Another possibility for measuring performance would be various string distances, such as the Levenshtein distance. These can be adapted to tokens instead of characters and do not have the alignment problem. We do not consider this further but note that relying on string

distance measures would only improve the metrics for our models.

Setup and training. We train separate models for dates and ages but both are based on the same attention neural network. Below we detail the choices of dictionary, hyper-parameters, and the training procedure used to train the two models on the type B death certificates.

Dictionary. The age transcription model uses a dictionary consisting of the digits $(0, 1, ..., 9)$ together with start and end markers to delimit the sequence. The final age dictionary is $\Omega_{age} = \{\texttt{<Start>}, \texttt{<0>}, \texttt{<1>}, ..., \texttt{<9>}, \texttt{<End>}\}$ which can represent integer ages of arbitrary length, in principle allowing for 3-digit ages. For example, 12 would be tokenised as

```
<Start>, <1>, <2>, <End>
```

The dictionary excludes ratios of years, e.g. 1/2, and any suffixes such as years, months, days, and hours. It would be straightforward to add additional tokens but since our ground truth transcriptions do not contain this information we cannot train the model to recognise them.

Description	Tokens
Day and year digits	<0>, <1>, <2>, <3>, <4>, <5>, <6>, <7>, <8>, <9>
Months	<January>, <February>, <March>, <April> <May>, <June>, <July>, <August>, <September> <October>, <November>, <December>
Separators	<DayMonthSeparator>, <MonthYearSeparator>
Sequence markers and padding	<Padding>, <Start>, <Stop>

Table A.3: Dictionary Ω_{date} for the date transcription model.

The date transcription model uses a larger dictionary. Each month is considered a separate token, <January>, <February>, ..., <December>, and all digits are separate tokens, <0>, <1>, ..., <9>. We include two separator tokens, <DayMonthSeparator> and <MonthYearSeparator>, which encode the separators between the day–month and month–year components of the dates. We also include sequence delimiters <Start>, <End> for

marking the start/end of the sequence and padding `<Padding>`. The entire dictionary Ω_{date} is given in Table A.3 and it provides a standardised representation of any date. For example, the date `1/7-2010` would be tokenised as

```
<Start>,<1>,<DayMonthSeparator>,<July>,<MonthYearSeparator>,<2>,<0>,<1>,
<0>,<End>
```

Similarly, the date `September 20th, 90` would by tokenised as

```
<Start>,<2>,<0>,<DayMonthSeparator>,<September>,<MonthYearSeparator>,
<9>,<0>,<End>
```

Data. The date model is trained on 11,630 manually transcribed dates. The age model is trained on 11,072 manually transcribed ages. To boost the amount of training data, we apply an augmentation procedure to create multiple distorted versions of each original training sample. Prior to training, the mean and variance are estimated using a sample of $20,000$ augmented images. The mean and variance are then used to standardise the pixel values in each training image. The ground truth sequences (respectively for dates and ages) are tokenised according to the defined dictionaries and they are padded to all have same length T. For age this length is $T = 4$ tokens while it is $T = 11$ tokens for dates. During training the models are fed batches of pairs $\{(\mathbf{x}_i, y_i)\}_i$ where each training pair consists of a field image $\mathbf{x}_i$ and its corresponding ground truth sequence y_i.

Hyperparameters. The transcription models require various hyperparameters that must be selected by the researcher. Some parameters are intuitive, such as. the size of the input images, but others are more abstract, such as the number of hidden nodes in a specific layer of the neural network. There is limited guidance for choosing these parameters. For the best performance, it is necessary to do hyperparameter optimisation where several sets of parameters are evaluated on a tuning dataset. We do not consider this and instead use manually selected values that yield acceptable results on the death certificates.

The feature network is the convolutional part of a pre-trained ResNet-101 network (He, Zhang, et al., 2016), which we tune during training with a learning rate of 0.0003. The

encoded image size is $(6, 20)$, which is obtained by applying adaptive max pooling after the final layer of the ResNet-101 feature network. The learning rate is 0.0005 for the remaining parameters (i.e. all parameters not in the feature network). The learning rate is lower for the feature network to combat over-fitting as the ResNet-101 network is deep. In addition, we use learning rate decay with a factor of 0.25 which is applied to both the main and feature network learning rates at steps $10,000$, $15,000$ and $25,000$.

The RNN cell is a Gated Recurrent Unit (GRU) (Cho et al., 2014). The number of hidden units in the attention network is 512 while it is $2,056$ in the RNN cell. The initial hidden state of the RNN cell is learned from the features using a fully-connected layer. We apply dropout with probability 0.5 before the final softmax prediction layer. The model is trained with back-propagation using the Adam optimiser (Kingma and Ba, 2014). The batch size is 42 images which is the maximum possible on our hardware.

The date model is trained for 45,000 steps until stagnating improvement in the loss function. This is equivalent to the model encountering each training sample on average 248 times (also known as 248 epochs). The age model is trained for $28,000$ steps implying that each sample is encountered on average 162 times. Note that the training datasets are augmented; while the model does encounter the same original sample multiple times, the sample will have different random distortions.

B. TRANSKRIBUS FOR DATE TRANSCRIPTION

To compare our ML pipeline with an off-the-shelf tool, we perform a comparison with Transkribus. Here, we use Transkribus to transcribe the birth and death dates from our dates evaluation dataset (1,000 dates in total), and compare the performance achieved by Transkribus with the performance of our ML pipeline. This step of our pipeline operates on the cropped images obtained after segmentation, but these are not usable with the current version of Transkribus. Transkribus is targeted at finding, segmenting, and transcribing text regions, which we found it completely unable to do using our cropped images. Furthermore, the table structure of the death certificates makes regional and baseline identification difficult, especially considering the mix of machine-printed and handwritten text combined with various scribbled remarks on the edges of many death certificates. Transkribus does not function on the cropped images and letting it work on the full death certificates makes it difficult to extract only the birth and death date transcriptions. To use Transkribus, we therefore transcribe the full death certificates and manually extract the transcribed birth and death dates from Transkribus. Potentially, this could be automated by comparing the coordinates of the date region to the coordinates of the text transcribed by Transkribus, followed by cleaning the extracted transcribed text. We compare the transcribed birth and death dates to our test sample to obtain the sequence accuracy of Transkribus, which is 73.9% (compared to 90.5% of our method). Note, however, that to use Transkribus to automatically transcribe the documents, similar to what our pipeline is able to, it would be required to adopt a method to extract only the relevant parts of the transcription made by Transkribus. In addition, one has to take into account the cost of transcription using Transkribus which quickly becomes very expensive when transcribing hundreds of thousands or even millions of documents with a pricing of 0.24 € per handwritten page when buying credits for 24,000 documents at a time (see https://readcoop.eu/transkribus/credits/). The relatively poor performance of Transkribus compared to our pipeline, combined with the large costs when using Transkribus to transcribe large collections of documents, motivate our custom

approach. Although Transkribus effectively handles transcription of many forms of streams of text, it is less well-suited for transcribing large collections of documents with a tabular structure, which is our focus.