

# The AI processes of HDAR

Version 1

November 2024

Amalie Regitze Faber Mygind



## Table of content

<b>Introduction.....</b>	<b>3</b>
<b>Data.....</b>	<b>3</b>
<b>Documentation.....</b>	<b>8</b>
.json.....	8
.csv.....	14
<b>Layout sorting .....</b>	<b>18</b>
Binary classification – death certificates .....	19
Multiclass classification .....	20
Subclass classification.....	21
Binary classification – coroner's certificates <1931 .....	22
<b>Segmentation.....</b>	<b>23</b>
Death certificates issued by doctors .....	23
'Ligsynsmandsattester' .....	24
Minipics .....	25
<b>Transcription.....</b>	<b>27</b>
<b>Cause of death code .....</b>	<b>30</b>
Segmentation and transcription .....	33
Linking.....	34
<b>Coding occupation with HISCO .....</b>	<b>35</b>
<b>Compute facilities.....</b>	<b>36</b>



## Introduction

In the period September 2023 to August 2024, selected variables from the Danish Health Authority's collection of death certificates from 1915 to 1943 have been digitized as part of work package 4 (WP4) in the MGR project. The purpose of the digitization is partly to be able to confirm deaths for use in the MGR project, and partly to be able to take the current digital cause of death register DAR back in time, and thus create a historical cause of death register HDAR. Note that in the finished HDAR only certificates from 1920-1943 is included.

This document describes the work on the actual digitization using artificial intelligence, but does not describe the data cleaning and processing that follows as the next step in the project.

## Data

The data consists of 1.572.734 retro-digitized pages of the Danish National Board of Health's death certificates from 1915 to 1943, including, among other things, back pages, box fronts, leaflets and coroner's certificates. Of these, 197.630 pages are included in a dataset (hereinafter referred to as DB2) where the content has previously been manually annotated via the National Archives' CS portal by volunteers. This dataset has served as the basis for training data for the majority of the models for HTR (handwritten text recognition) that have been trained throughout the project. The remaining 1.375.104 pages constitute the dataset that will henceforth be called DB345.

The variables (relevant to the project) that had been manually transcribed for the DB2 dataset were name, age, date of birth, date of death, marital status, occupation and cause of death. For the remaining variables in HDAR which were not included in the manual transcriptions from the DB2 set – primary illness and manner of death – we have generated training data through the project. This has been done using student assistance and through voluntary efforts.


Which variables must be read for a death certificate depends on the specific type of death certificate in question. The Danish Health Authority's death certificates consist of 9 different types throughout the period, each with a slightly different layout. In addition, there are death certificates completed by coroners, which until 1931 have a completely different layout than the remaining death certificates, but only one single layout throughout the entire period. Table 1 shows which variables have been read from the different types of death certificates during the HDAR project. Table 2 gives a brief description of the content of the 9 different layout types, and shows examples of death certificates for each layout type, coroner's certificates and other non-relevant scanned material (hereinafter referred to as "slat").





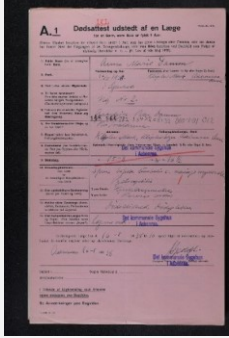
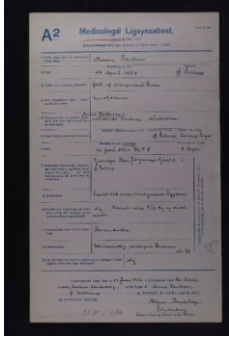
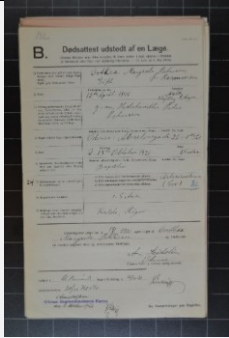
Variable Layout type	Name	Birthdate	Deathdate	Cause of death	Primary illness	Occupation	Marital status	Manner of death	Age
A	X	X	X	X		X			
A1	X	X	X	X	X	X			
A2	X	X	X	X		X		X	
B	X	X	X	X		X	X		
B1	X	X	X	X	X	X	X		
B2	X	X	X	X		X	X	X	
C		X		X		X			
D	X	X	X	X		X			
E	X	X	X	X		X	X		
Ligsynsmand	X		X	X		X	X		X

Table 1: Overview of which variables are read from the different layout types of death certificates. An X symbolizes that the variable is read.

Layout type	Description	Period in use	Count in DB2	Count in DB345	Example
A	Death certificate issued by a doctor for a child under 1 year old	1915-1930	10030	56319	

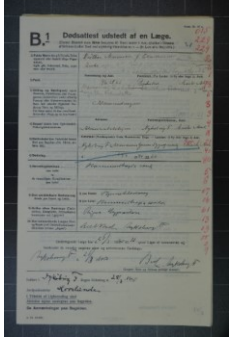
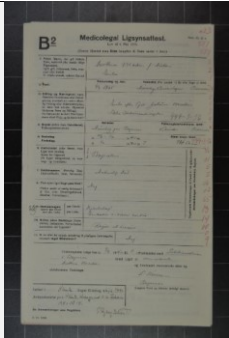
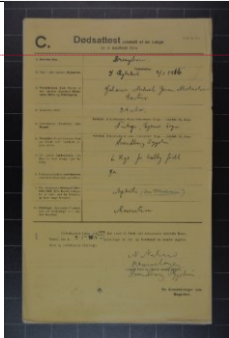




Layout type	Description	Period in use	Count in DB2	Count in DB345	Example
A1	Death certificate issued by a doctor for a child under 1 year old	1931-1943	9967	39108	
A2	Medicolegal death certificate for a child under 1 year old	1931-1934	1173	4583	
B	Death certificate issued by a doctor	1919-1930	60854	323708	



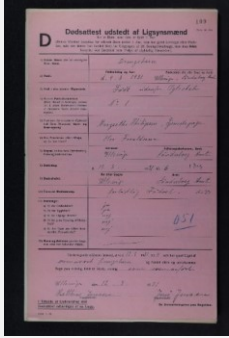
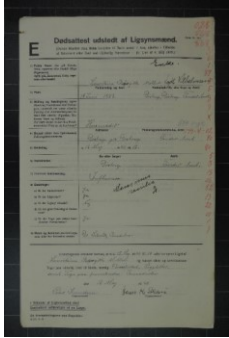
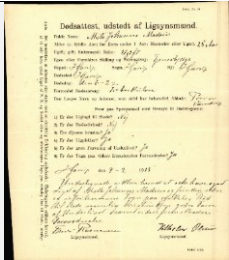


Layout type	Description	Period in use	Count in DB2	Count in DB345	Example
B1	Death certificate issued by a doctor	1931-1943	82380	327507	
B2	Medicolegal death certificate	1931-1943	9805	44893	
C	Death certificate issued by a doctor for a stillborn child	1915-1931, 1938-1943	4039	14179	

Kommenterede [JR1]: Ingen C'er attester mellem 1932-1937. Kasseret hos Sundhedsstyrelsen?





Layout type	Description	Period in use	Count in DB2	Count in DB345	Example
D	Death certificate issued by a coroner for a child under the age of 1 from 1931 onwards	1931-1943	611	2217	
E	Death certificate issued by a coroner from 1931 onwards	1931-1943	4612	8457	
Ligsynsmands-attest	Death certificate issued by coroner primarily before 1931	1915-1943	10259	37713	
Slat	Non-relevant images scanned in connection with the retro-digitization of the collection of death certificates.	N/A	3900	516419	





Layout type	Description	Period in use	Count in DB2	Count in DB345	Example
	This includes back pages, box covers, pages with leaflets, blurred pages, etc.				 

Table 2: Description of the content of and examples of the different layout types that make up the Danish National Board of Health's death certificates in the period 1915-1943.

Figure 1 shows a full overview of the number of certificates for each layout type for each year in the period. In the overview, both datasets DB2 and DB345 are combined..

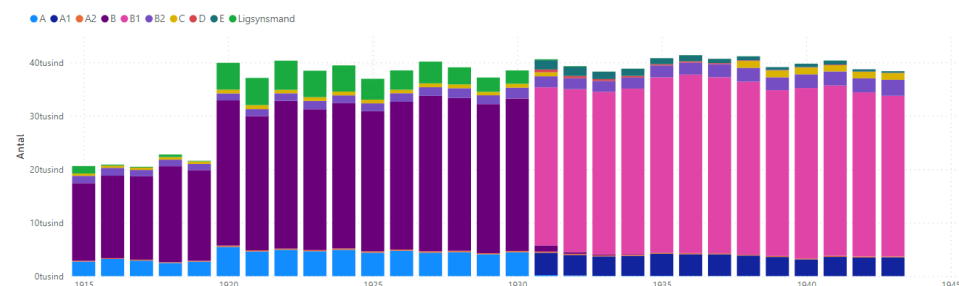


Figure 1: Overview of the number of certificates for each layout type for each year in the period for the total data volume, i.e. the combination of the DB2 and DB345 sets. Each color represents a different layout type..

All images in the project are freely available on ArkivalierOnline<sup>1</sup> and has been downloaded from there.

## Documentation

### .json

For all images that are to be processed and read, a .json file is created. This stores information about and results of all model runs that have been performed on the image in order to document the entire process. As further described in the section on layout sorting below, all irrelevant images from the collection are sorted out as the first step in the process. A .json file is not created for these sorted out images. For all images where a

<sup>1</sup> <https://arkivalieronline.rigsarkivet.dk/da/other/other-collection/26>





.json file is created, the .json file is named the same as the image, but with the .json extension instead of the image's file extension.

Table 3 shows an example of the content of a .json file for an image of a death certificate and an explanation of the different variables in the file. The example is an example of an image where all variables are included. For some images, some models will not have been run and these models will therefore not appear in the .json file. An example of this is death certificates for stillborn children which do not contain a field with the main disease. Therefore, a reading of the main disease has not been run on these certificates, and the .json file for the images will therefore not contain information about the model for the main disease.

Variable	Value	Description
file_path	E:\\HDAR\\DødsattestBilleder4\\Mappe_0\\37771888.jpg	The path to the image location on the hard drive
binary_class	Attest	Result of the binary classification
binary_class_probability	0.9999303817749025	Confidence score for the result of the binary classification
binary_model	ViT	The name of the architecture for the binary classification model
binary_model_path	D:/Work/DAR/binaryclassification/models_retrain/ViTbinaryclass_model_98.8	The path to the file with the saved binary classification model
binary_pred_time stamp	26-09-2023_11:17	Date and time of running the binary classification



		model on the image
multi_class	B1	Result of multiclass classification
multi_class_probability	0.999991774559021	Confidence score for the result of multiclass classification
multi_model	SegformerForImageClassification	The name of the architecture for the multiclass classification model
multi_model_path	D:\\Work\\DAR\\multiclassification\\Segformer\\models_retrain\\Segformermulticlass_model_99.2	The path to the file with the saved model for multiclass classification
multi_pred_timestamp	26-09-2023_11:18	Date and time of running the multiclass classification model on the image
sub_class	1.0	Result of subclass classification
sub_class_probability	0.9996475	Confidence score for the result of subclass classification
sub_model	ViT	The name of the architecture for the subclass





		classification model
sub_model_path	D:\\Work\\DAR\\submulticlassification\\B1\\models_retrain\\ViTsubclass_model_98.9	The path to the file with the saved model for subclass classification
sub_pred_timestamp	26-09-2023_22:44	Date and time of running the model for subclass classification on the image
segmentation_model	YOLOv8n	The name of the architecture of the model for segmentation
segmentation_model_path	D:\\Work\\DAR\\segmentation\\object_detection\\runs\\detect\\DAR_yolo_1280_batch16_100e\\weights\\best.pt	The path to the file with the saved model for segmentation
segmentation_timestamp	09-03-2024_06:31	Date and time of running the segmentation model
intersection_points	[ { "row_idx": 0, "points": [ 909.0333862304688, 1010.8751220703125 ] }, ]	Set of x and y coordinates for each intersection detected by the segmentation model. This is further described in the segmentation section.



	<pre>{   "row_idx": 1,   "points": [     1192.549560546875,     1009.0062866210938   ] }, ... ]</pre>	
cell_points	<pre>[   {     "row_idx": 0,     "points": [       909.0333862304688,       1010.8751220703125,       1192.549560546875,       3022     ]   },   {     "row_idx": 1,     "points": [       1192.549560546875,       1009.0062866210938,       1379.056640625,       3022     ]   },   ... ]</pre>	Set of coordinates for the upper right and lower left corners of the individual table cell, calculated based on the detected intersection points. This is further described in the section on segmentation .





htr	<pre>"name": {   "prediction": {     "top1prediction": "Rasmus Rasmussen Stougaard",     "top1probability": 0.305322,     "top2prediction": "Rasmus Rasmussen Stougaard",     "top2probability": 0.117773,     "top3prediction": "Rasmus Rasmussen Staugaard",     "top3probability": 0.138327,     "top4prediction": "Rasmus Rasmussen Stengaard",     "top4probability": 0.062352,     "top5prediction": "Rasmus Rasmussen Hougaard",     "top5probability": 0.137294   },   "basemodel": "TrOCR-large-handwritten",   "model_checkpoint": "D:\\Work\\DAR\\htr\\names\\model_finetuned_from_u cloud\\names_model_3.92",   "prediction_timestamp": "02-09-2024_02:35" }, "birthdate": {   "prediction": {     "top1prediction": "06011853",     "top1probability": 0.9999,     "top2prediction": "05011853",     "top2probability": 0.8288,     "top3prediction": "08011853",     "top3probability": 0.819,     "top4prediction": "06041853",</pre>	<p>Results of all HTR models that have been run on the image. Here is an example of the results from the reading of the name field and the birthday field. For each reading, the top 5 most likely readings are stored together with their associated confidence score. In addition, the name of the architecture for the model, the path to the model location on the hard drive, and the time of the model run are stored.</p>
-----	--	--





	<pre>"top4probability": 0.7496, "top5prediction": "16011853", "top5probability": 0.744 }, "basemodel": "TrOCR-large-handwritten", "model_checkpoint": "D:\\Work\\DAR\\htr\\birthdates\\model_finetuned_fro m_ucloud2\\birthdate_model_1.07", "prediction_timestamp": "25-03-2024_10:25" }, ... ]</pre>	
--	--	--

Table 3: Example of the content of a .json file for an image of a death certificate and explanation of the different variables in the file.

### .CSV

For each variable read with HTR and for classification of all images, the results of the model runs are collected in an associated .csv file. These csv files serve as input to the combined database that, after data cleaning and possible manual correction, will constitute HDAR.

The csv files for classification contain 16 variables, which are described with two examples from the DB345 dataset in Table 4 below.

Variable	Example 1	Example 2	Description
FilSti	E:\\HDAR\\ DødsattestBilleder3\\ Mappe_0\\73704224.jpg	E:\\HDAR\\ DødsattestBilleder3\\ Mappe_15\\70148565.jpg	The path to the image location on disk
Binary Class	Slat	Attest	Result of running the binary classification model with the outcomes [Slat, Attest]
Binary Prob	0.9999876	0.99994624	Confidence score for the result of the





			run with the binary classification model
Binary Model	ViT	ViT	The name of the architecture of the binary classification model
Binary Model Path	D:/Work/DAR/binaryclassification/models_retrain/ViTbinaryclass_model_98.8	D:/Work/DAR/binaryclassification/models_retrain/ViTbinaryclass_model_98.8	The path to the binary classification model location on disk
Binary Pred Time	26-09-2023_11:17	26-09-2023_11:17	Date and time of running the binary classification model on the image
Multi Class	Null	B2	Result of running the multiclass classification model with the outcomes [A, A1, A2, B, B1, B2, C, D, E]
Multi Prob	Null	0.9999989	Confidence score for the result of the run with the multiclass classification model
Multi Model	Null	SegformerForImageClassification	The name of the architecture of the multiclass





			classification model
Multi Model Path	Null	D:\Work\ DAR\ multiclassification\ Segformer\models_retrain\ Segformermulticlass_model_99.2	The path to the multiclass classification model location on disk
Multi Pred Time	Null	26-09-2023_11:18	Date and time of running the multiclass classification model on the image
Sub Class	Null	0.0	Result of running the subclass classification model with the outcomes [0.0, 1.0]
Sub Prob	Null	0.9994305	Confidence score for the result of the run with the subclass classification model
Sub Model	Null	ViT	The name of the architecture of the subclass classification model
Sub Model Path	Null	D:\Work\ DAR\ submulticlassification\ B2\models_retrain\ ViTsubclass_model_100.0	The path to the subclass classification model location on disk
Sub Pred Time	Null	27-09-2023_00:24	Date and time of running the model for subclass





			classification on the image
--	--	--	--------------------------------

Table 4: Two examples of the content of a .csv for classification for a slat image and an image of a death certificate, respectively, as well as an explanation of the different variables in the files.

The CSV files for HTR readings contain 11 variables, which are described with two examples of readings of marital status from the DB345 dataset in Table 5 below. For each HTR reading, the model provides 5 suggestions for a reading with the associated confidence score.

Variable	Example 1	Example 2	Description
FilSti	E:\HDAR\ DødsattestBilleder4\ Mappe_0\ 37771888.jpg	E:\HDAR\ DødsattestBilleder4\ Mappe_0\ 37771889.jpg	Path to the image location on disk
Top1Text	Enke	Gift	The model's best guess for a prediction
Top2Text	Ugift	Fraskilt	The model's second best guess for a prediction
Top3Text	Gift	Ugift	The model's third- best guess for a prediction
Top4Text	Frasret	Enke	The model's fourth- best guess for a prediction
Top5Text	Frke	Fraskift	The model's fifth best guess for a prediction
Top1Prob	1.0	1.0	Confidence score for the model's best guess
Top2Prob	0.6359	0.6769	Confidence score for the model's second best guess
Top3Prob	0.4973	0.6374	Confidence score for the model's third-best bid



Top4Prob	0.4170	0.5313	Confidence score for the model's fourth-best bid
Top5Prob	0.3933	0.4790	Confidence score for the model's fifth-best bid

Tabel 5: Two examples of the content of a .csv for HTR reading of civil status for two images of death certificates, as well as an explanation of the different variables in the files.

For HTR readings of names and birthdays, in addition to the 11 variables mentioned in Table 5, each of the 5 predictions/bids is also stored divided into individual tokens with associated confidence scores for each token. This is only done for names and birthdays, as these two variables are the most essential for linking an individual from a death certificate to the CPR or MGR. By specifying a confidence score for each token in a prediction instead of just one overall confidence score, uncertain parts of a prediction can be identified, and hopefully facilitate linking to the CPR or MGR.

## Layout sorting

Among the approximately 1.5 million retro-digitized pages, there is a large amount of images of back pages, leaflets, etc. that do not contain relevant information. These are sorted out using binary classification, where all images are either assigned to the class "certificate" or "slat". Initially, coroner's certificates from before 1931 are also sorted out as slat, since the layout of these differs from ordinary death certificates to such an extent that they must be treated separately.

For all images classified as a certificate, multi-class classification is performed, where the image is assigned to one of the 9 layout classes. For 5 of the 9 layouts - A1, A2, B1, B2 and C - a small but significant difference in layouts within the class itself is identified.

Therefore, another round of binary classification is performed for the images in each of these 5 layout classes, where the images are divided into subclasses A10 and A11, A20 and A21, B10 and B11, B20 and B21, and C0 and C1 for the A1, A2, B1, B2 and C layout classes, respectively.

It is worth noting that even though the layouts used clearly depend on the year of death, and we have good metadata for the year of death, this is not enough to be able to clearly distinguish between subclasses. This is because doctors may have excess death certificates from last year (and last year's layout) lying around, and thus have them filed away "incorrectly" after use. This illustrates the importance of not only relying on metadata, but making sure to do classification on the full material.

Later, another round of binary classification is performed where the original residual pile consisting of "slat images" is either assigned to the class "coroner's certificate" or "slat" to identify the coroner's certificates.

As a starting point, all models in the project are trained with data from the DB2 dataset unless otherwise explicitly stated.



## Binary classification – death certificates

To perform binary classification for layout sorting, a transformer-based model with a ViT (Vision Transformer)<sup>2</sup> backbone and a linear classification head from the Transformers library is used. The reason for this choice of model architecture is that we have had good experience with this model for binary classification from the past. The ViT model used as the backbone is loaded from the checkpoint google/vit-large-patch16-224<sup>3</sup> via Hugging Face. The classification head, which consists of a single linear layer with softmax, is randomly initialized.

The processor used to preprocess and tokenize the images is ViTImageProcessor<sup>4</sup> from the Transformers library. The processor is loaded from the same checkpoint as the ViT model via Hugging Face. This processor is used for all models for binary classification throughout the project unless otherwise explicitly stated.

Three rounds of training are performed with a human-in-the-loop approach. In the first round, a dataset consisting of 291 slat images and 35\*9=315 attest images is used. From this, 25 slat images and 35 attest images are reserved for a dedicated test set. A first round of training is performed, after which inference is run with the trained model (model1) on the full dataset consisting of 1.5 million images. From here, the 2500 images with the lowest classification probability are selected and their layout type is annotated manually. The 2500 images are now included together with the original training data in a new dataset, which is used to train the binary classification model (model2) from scratch. The training is repeated, and after training with the new training data, an accuracy of 99% is achieved on the test set.

In the third round, all the images from the DB2 set that have been classified as slat by model2, and which have later also been classified as slat by the binary classification model for identifying coroner's certificates, are inspected. This involved approximately 6000 images, which was a manageable amount to review manually. Based on this manual inspection, it was discovered that model2 had a 25% false negative rate, which means that 25% of the images from the DB2 set that had been classified as slat by both binary classification models were actually real death certificates. These approximately 6000 images were therefore manually annotated and added to the total amount of training data. In addition, the 1500 images with the lowest confidence score, the 500 images with the highest confidence score, and 500 images in the middle of the confidence score spectrum were taken for running with model2 on the DB2 and DB345 sets, respectively. These images were also manually annotated and added to the total amount of training data. Training was done again with the total amount of training data, and then with this new model (model3) inference was run on all images classified as slat by both classification models for both the DB2 and DB345 data sets (approximately 500,000 images). This resulted in an additional approximately 2200 and approximately 800 death certificates

---

<sup>2</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/vit#transformers.ViTForImageClassification](https://huggingface.co/docs/transformers/en/model_doc/vit#transformers.ViTForImageClassification)

<sup>3</sup> <https://huggingface.co/google/vit-large-patch16-224>

<sup>4</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/vit#transformers.ViTImageProcessor](https://huggingface.co/docs/transformers/en/model_doc/vit#transformers.ViTImageProcessor)



being correctly classified for the DB2 set and the DB345 set, respectively. Inference was only run on these “double-slat” images with model3 and not on the total amount of data, as the manual inspection did not show any significant rate of false positives, only false negatives.

For all three trainings, an AdamW optimizer from PyTorch is used with a fast learning rate of  $5e-5$ , which has proven to be the best choice of optimizer and learning rate to finetune this type of model from. As a loss function, CrossEntropyLoss from PyTorch is used with class weights to compensate for an unbalanced dataset. A batch size of 16 and a validation split of 20% are used. Training is carried out for up to 30 epochs with early stopping pba. validation accuracy with a patience of 2 epochs. These training parameters have been used for all models trained for binary classification throughout the project unless otherwise explicitly stated.

For all images that are classified as a certificate, a .json file is created that is named the same as the image, but with the .json extension. All information and results for all models that have been run on the image in question are stored here as described in the documentation section above.

### Multiclass classification

To perform multiclass classification – to sort between the 9 layouts – a SegformerForImageClassification<sup>5</sup> model from the Transformers library is used. The last layer in the model is a linear layer, whose size is set to 9 in order to classify between the 9 different layouts. The model is initiated from the checkpoint nvidia/mit-b5<sup>6</sup> via Hugging Face. The reason for this choice of model architecture is that we have previously had good experiences with this model for multiclass classification

From each image classified as a death certificate of the binary classification, the upper left corner is cut out and used as input to the model. For preprocessing and tokenization, AutoImageProcessor<sup>7</sup> from the Transformers library is used, which is loaded from the same checkpoint as the model. 1000 images of each layout class are used as training data, except for layout class D, where only 590 images are used. Of these, 100 images of each layout class are reserved for a dedicated test set.

The AdamW optimizer from PyTorch is trained with a fast learning rate of  $6e-5$ , which has proven to be the best choice of optimizer and learning rate to finetune this model from. The cross entropy loss function built into the model is used as the loss function. The training is performed with a batch size of 8 and a validation split of 20%. As for the binary classification model, the training is performed for up to 30 epochs with early stopping based on validation accuracy, and with a patience of 2 epochs. After training, an accuracy of 98.9% is achieved on the test set.

---

<sup>5</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/segformer#transformers.SegformerForImageClassification](https://huggingface.co/docs/transformers/en/model_doc/segformer#transformers.SegformerForImageClassification)

<sup>6</sup> <https://huggingface.co/nvidia/mit-b5>

<sup>7</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/auto#transformers.AutoImageProcessor](https://huggingface.co/docs/transformers/en/model_doc/auto#transformers.AutoImageProcessor)

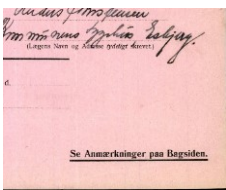
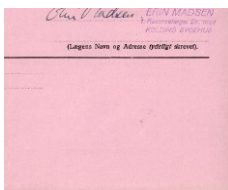

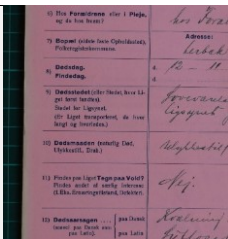

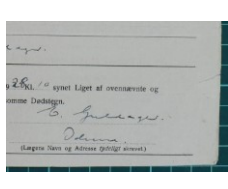




## Subclass classification

The same pretrained ViT model is used as for the initial layout sorting and with the same randomly initialized classification head. A separate model is trained for each of the 5 layouts that have sublayouts.

The same preprocessing is performed for training each of the 5 models, except for which area of the page is cropped out and used for training. Table 6 below shows an example of the cropped area for each subclass used for training for each of the 5 models.

	Subclass 0	Subclass 1	Description of difference
<b>Multiclass A1</b>			The order of the fields for place of death and date of death is different. For A10, the order is date of death-place of death. For A11, it is place of death-date of death. For A10, "See Notes on Back" is printed in the lower right corner. For A11, the lower right corner is blank.
<b>Multiclass A2</b>			A20 contains 13 rows. A21 contains 14 rows. For A20, row 8 is higher than for A21.
<b>Multiclass B1</b>			The order of the fields for place of death and date of death is different. For B10, the order is date of death-place of death. For B11, it is place of death-date of death. For B10, "See Notes on the Back" is printed in the lower right corner, which is not the case for B11.





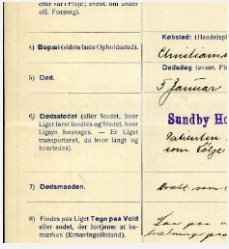
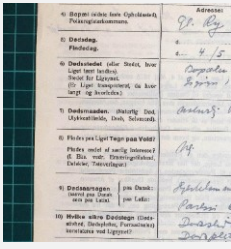
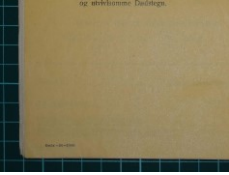
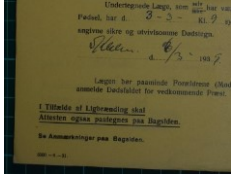
	Subclass 0	Subclass 1	Description of difference
<b>Multiclass B2</b>			For B20, row 6 is higher than for B21.
<b>Multiclass C</b>			C0 contains 10 rows where C1 contains 11 rows. For C1 there is preprinted text in the lower left corner. For C0 the corner is empty.

Table 6: Examples of subclasses for each of the 5 layout classes with subclasses. For each multiclass, an example of each subclass is shown as well as a description of how the two subclasses differ from each other.

The training approach is the same for the 5 models and is the same as for the initial layout sorting. Training is again done with a human-in-the-loop approach. In the first round of training, 500 images are used for training, of which 50 are reserved for the test set. After the first training, inference is run on all images in the layout class in question. All images with a classification probability of 70% or less are manually annotated and included as training data in the next round of training.

After training, an accuracy of 98% is achieved on the test sets for A2 and B2 and of 100% for A1, B1 and C.

### Binary classification – coroner's certificates <1931

The binary classification to identify the pre-1931 coroner's certificates is performed with the same ViT-based model as for the other binary classifications. For training data, 778 images of coroner's certificates and 616 images of slat/other are manually identified from the stack of images that were originally classified as slat. From the total training dataset of 1394 images, 100 images are reserved for a dedicated test set.

This model is trained in only one pass. The full images and the same preprocessing as for the other binary classification models are used. After training, a classification accuracy of



100% is achieved on the test set. By inferring the full slat stacks, 10,259 and 37,713 coroner's certificates are identified for the DB2 and DB345 sets, respectively, i.e. a total of 47,972 coroner's certificates with the special layout from before 1931.

## Segmentation

### Death certificates issued by doctors

To segment the death certificates into meaningful smaller components, the schema structure of the certificates is utilized. The intersections between vertical and horizontal lines are identified using the object detection model YOLO, which makes it possible to segment each individual schema cell into its own minipic.

A YOLOv8 model is trained from the Ultralytics library<sup>8</sup>. The starting point is the pre-trained nano model and fine-tuned with a training dataset consisting of 198 death certificates with manually annotated intersections. Training is performed in 100 epochs with a batch size of 16 and by resizing the images to 1280 pixels in the largest dimension.

During inference, each death certificate is first run through YOLO and then through a series of customized "cleaning" functions. These functions ensure that the correct intersections are identified, that outliers are removed, and that any missing intersections are inserted in the correct locations. To perform this cleaning of the intersections, a template of the expected location of the intersections has been manually created for each layout class. The cleaning functions compare the relative distances and locations between the intersections in the template with the found intersections and can remove and insert intersections from this until they match the template.

In the example in Figure 2, the YOLO model has only been able to identify the top five intersections, which is shown with blue crosses in the left image. The red crosses in the left image show the points from the template that belong to the layout class of the image. The right image in Figure 2 shows the final intersections with red crosses. These are the result of the cleaning functions where the YOLO output and the template are entered, and through which the location of the missing intersections has been found.

---

<sup>8</sup> <https://github.com/ultralytics/ultralytics>



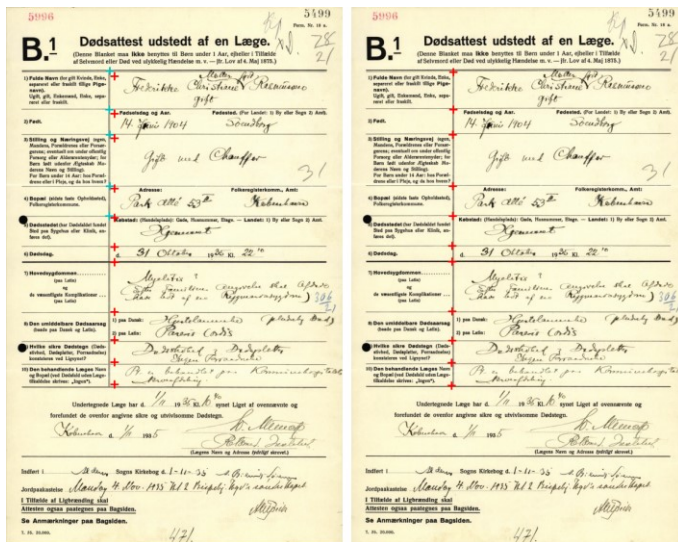


FIGURE 2: EXAMPLE OF SEGMENTATION OF A DEATH CERTIFICATE. ON THE LEFT IS THE RAW OUTPUT FROM THE YOLO MODEL WITH BLUE CROSSES AND THE TEMPLATE FOR THE IMAGE LAYOUT CLASS SHOWN WITH RED CROSSES. ON THE RIGHT IS THE RESULT OF THE CLEANING WHERE THE FOUND CROSSES TOGETHER WITH THE TEMPLATE ARE USED TO INFER THE LOCATION OF THE MISSING CROSSES.

The coordinates of the final intersection points for each death certificate are stored in the associated .json file.

### 'Ligsynsmandsattester'

For the coroner's certificates from before 1931, there is not the same schema structure to lean on as for the regular death certificates. Instead, the fixed structure of preprinted text on the page is used to identify the location of relevant information.

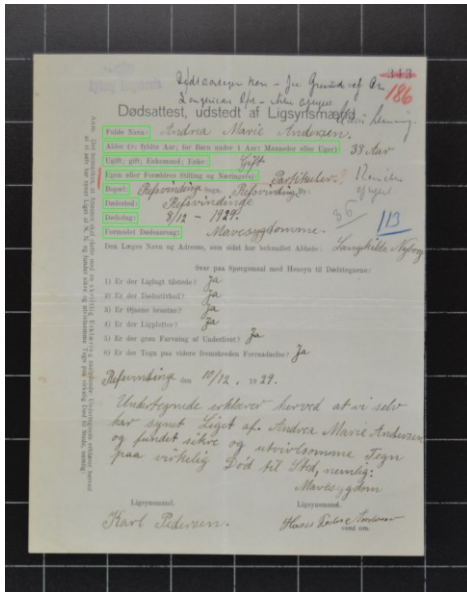
A YOLO model is trained again. This time to identify chunks of preprinted text. The model was trained in the same way as for the regular death certificates, but on a training dataset consisting of 100 manually annotated coroner's certificates. On the coroner's certificate in Figure 3, a green box is placed around the chunks of preprinted text that are identified by the fully trained YOLO model. Since all coroner's certificates between 1915 and 1931 have the same layout, it is always the same 8 chunks of text that must be identified and always in the same order. For the coroner's certificates where the YOLO model did not find the correct number of text chunks, the missing text chunks and their size were inferred from







the location and relative distance between the found chunks through custom-made functions.



**FIGURE 3: EXAMPLE OF A CORONER'S CERTIFICATE FROM BEFORE 1931. PIECES OF PREPRINTED TEXT IDENTIFIED WITH THE YOLO MODEL ARE OUTLINED WITH GREEN BOXES.**

As with regular death certificates, the coordinates from the segmentation of each coroner's certificate are stored in the associated .json file.

## Minipics

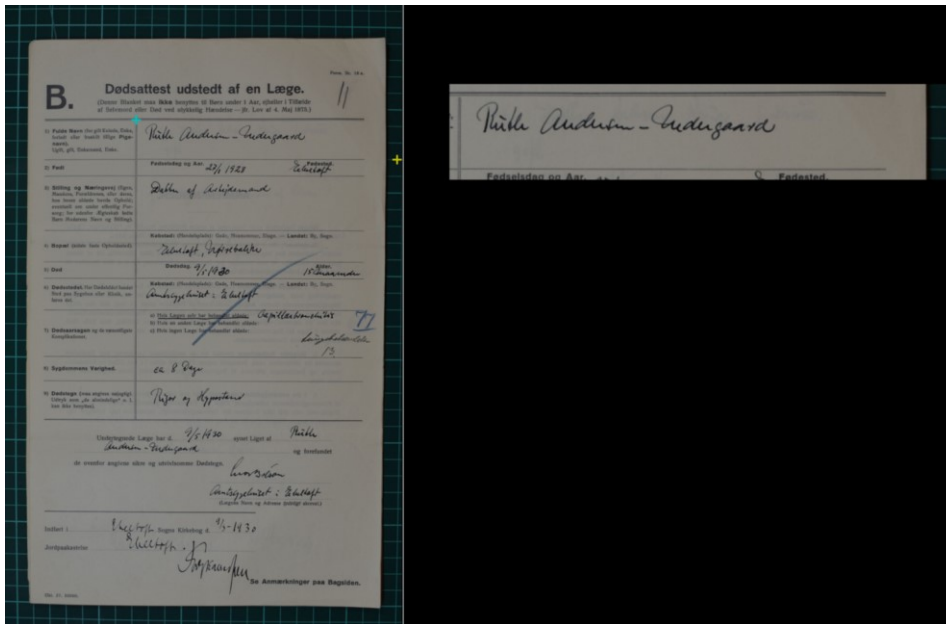
For each death certificate and each coroner's certificate, a .json file with segmentation coordinates has been saved after segmentation. How these segmentation coordinates are converted into relevant minipics differs depending on whether it is a regular death certificate or a coroner's certificate.

For regular death certificates, a minipic is defined based on the coordinate of its upper left corner and its lower right corner. The upper left corner for a given table field is taken directly from the intersection points found during segmentation and which are saved in the .json file. The lower right corner is defined by taking the same y-coordinate as the next intersection point in the row, and as the x-coordinate take the full width of the image minus 10 pixels. Since the intersection points saved in the .json file are sorted by increasing y-coordinate (i.e. from top to bottom), the index of the table field that you want to cut can be specified by using that index in the row of intersection points as the upper left coordinate. Figure 4 shows an example of the top left coordinate and bottom right





coordinate for the top table field on a regular death certificate, as well as the minipic that has been cut out based on this. The cutout is made with a margin of 25 pixels so that an extra frame of 25 pixels is cut in the directions up, down and to the left of the shown coordinates.

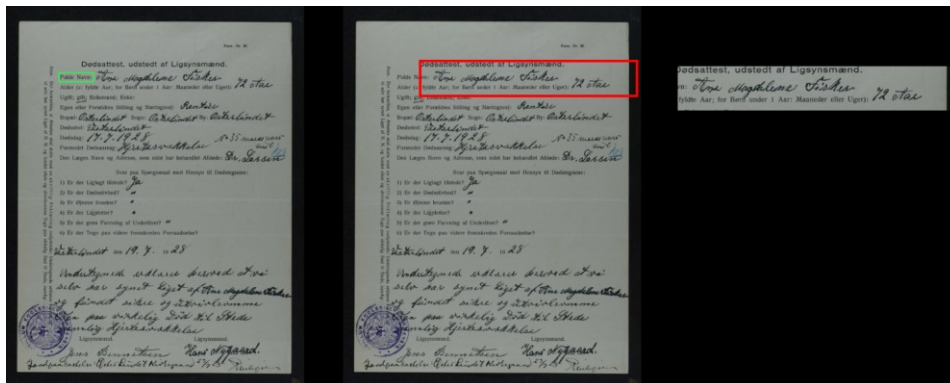


**FIGURE 4: EXAMPLE OF A TYPICAL DEATH CERTIFICATE WHERE THE TWO COORDINATES THAT DEFINE THE TOP TABLE FIELD ARE SHOWN. THE TOP LEFT COORDINATE, SHOWN IN BLUE, IS TAKEN DIRECTLY FROM THE LIST OF INTERSECTION POINTS FROM IMAGE'S .JSON FILE FROM THE SEGMENTATION. THE LOWER RIGHT COORDINATE (SHOWN IN YELLOW) IS COMPOSED OF THE Y-COORDINATE OF THE NEXT INTERSECTION POINT IN THE ROW AND THE X-COORDINATE AS THE IMAGE WIDTH MINUS 10 PIXELS. ON THE RIGHT IS THE CUT OUT MINIPIC.**

For the coroner's certificates <1931, text chunks of preprinted text are identified during segmentation as shown in Figure 3 and again in the left image in Figure 5. The upper and lower right corners of the green box surrounding the preprinted text chunk are the starting point for clipping the minipic. From here, go 25 pixels to the left and 1.5 times the height of the text chunk up and down, and this gives the coordinates for the upper and lower left corners of the red box seen in the middle image in Figure 5. From the two left coordinates, the red box is extended to the full width of the image minus 10 pixels. This results in the



red box in the middle image in Figure 5, and clipping is then carried out. The resulting minipic is seen on the right in Figure 5.



**FIGURE 5: EXAMPLE OF A CORONER'S CERTIFICATE <1931 AND THE PROCESS FOR CUTTING OUT THE MINIPIC OF THE TOP FIELD. ON THE LEFT, THE PRE-PRINTED TEXT PIECE IDENTIFIED DURING SEGMENTATION IS SHOWN WITH A GREEN BOX. IN THE MIDDLE, THE RED BOX IS INFERRED FROM THE GREEN BOX AND DEFINES WHERE TO CUT. ON THE RIGHT, THE FINAL MINIPIC OF THE NAME THAT HAS BEEN CUT OUT IS SHOWN.**

For the DB2 dataset, a minipic for each of the relevant variables for each death certificate/coroner's certificate has been saved and used to train transcription models. For the DB345 dataset, which was not used for training, no minipics have been saved. Instead, the relevant minipic has been clipped out as part of inference, and discarded again after the reading is complete..

## Transcription

As a starting point for all HTR in the project, the pre-trained TrOCR<sup>9</sup> model from the checkpoint microsoft/trocr-large-handwritten<sup>10</sup> from Hugging Face is used. For each variable to be read (see table 1), a model is fine-tuned based on this pre-trained model. The DB2 dataset is used as training data, from which minipics of the relevant fields are cut out based on coordinates from the segmentation. The size of the training dataset for the individual model depends on how many certificates with the relevant layout types are found in the DB2 set. For example, the variable 'cause of death' must be read for all layout types, while the variable 'main disease' is only read for layout types A1 and B1. The training dataset for the model for cause of death will therefore be larger than for the model for main disease. For some variables, including main disease, the training dataset is not large enough to achieve satisfactory precision by training directly from the aforementioned checkpoint. For this reason, some models are trained further from other models instead of

<sup>9</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/trocr](https://huggingface.co/docs/transformers/en/model_doc/trocr)

<sup>10</sup> <https://huggingface.co/microsoft/trocr-large-handwritten>



from the mentioned checkpoint. An example of this is the model for main disease that is trained further from the fully trained model for causes of death. Most models for reading coroner's certificates are also trained further from the models for reading ordinary death certificates. Table 7 shows which models are trained directly from the mentioned checkpoint and which are trained further from other fully trained models. All training data for all models is preprocessed and tokenized using the TrOCRProcessor processor<sup>11</sup> that is initiated from the same checkpoint as the pretrained model.

All TrOCR models in the project are trained with the same training parameters and the same training strategy: Of the total data volume, approximately 10% is reserved for a dedicated test set, on which the model is tested after training. Of the remaining data, 10% is used for validation during training. Training is carried out with a batch size of 8 and with the AdamW optimizer with a fast learning rate of 6e-5. The language modeling loss function built into the pretrained model is used as loss. Beam search with a beam width of 5 and a length penalty of 2 is used to generate predictions. Character error rate (CER) is used as metric. Training is performed for up to 100 epochs with early stopping with validation CER as criterion, and with a patience of 5 epochs.

For the coroner's certificates <1931, separate models are trained for each variable separately from the models for the regular death certificates, since the fields on the coroner's certificates visually differ so much from the fields on the regular death certificates.

After training, inference is run for each model on the dedicated test set. CER is used again as during training as the accuracy metric. The result of inference on the test set for each variable model, for both the regular death certificates and the coroner's certificates, is shown in Table 7.

Variable	Test CER	Certificate type	Trained from model	Count training-data	Count test-data	Count inference-data
Navn	3,92%	Almindelig dødsattest	microsoft/trocr-large-handwritten	167386	10000	820978
Fødselsdag (DDMMÅÅÅÅ)	1,07%	Almindelig dødsattest	microsoft/trocr-large-handwritten	176116	5120	820978
Dødsdag (DDMM)	0,58%	Almindelig dødsattest	microsoft/trocr-large-handwritten	172382	4881	806798

<sup>11</sup> [https://huggingface.co/docs/transformers/en/model\\_doc/trocr#transformers.TrOCRProcessor](https://huggingface.co/docs/transformers/en/model_doc/trocr#transformers.TrOCRProcessor)



Variable	Test CER	Certificate type	Trained from model	Count training-data	Count test-data	Count inference-data
Civilstand	3,63%	Almindelig dødsattest	microsoft/trocr-large-handwritten	144718	9575	704571
Dødsårsag	9,63%	Almindelig dødsattest	microsoft/trocr-large-handwritten	171380	10000	820978
Erhverv	10,54%	Almindelig dødsattest	Erhvervsmodel for ligsynsmands-attester	81515	7500	820978
Hovedsygdom	4,77%	Almindelig dødsattest	Dødsårsagsmodel for almindelige dødsattester	3888	151	92347 (DB2) + 366618 (DB345)
Døds måde	4,02%	Almindelig dødsattest	Hovedsygdoms-model for almindelige dødsattester	4671	151	10977 (DB2) + 49476 (DB345)
Navn	3,30%	Ligsynsmandsattest	Navnemodel for almindelige dødsattester	9226	1000	37713
Alder	2,63%	Ligsynsmandsattest	microsoft/trocr-large-handwritten	57886	1000	37713
Civilstand	1,78%	Ligsynsmandsattest	Civilstandsmodel for almindelige dødsattester	9226	1000	37713
Erhverv	12,02%	Ligsynsmandsattest	microsoft/trocr-large-handwritten	57924	1000	37713
Dødsdag (DDMM)	1,80%	Ligsynsmandsattest	Dødsdagsmodel for almindelige dødsattester	9226	1000	37713
Dødsårsag	8,22%	Ligsynsmandsattest	Dødsårsagsmodel for	56537	1000	37713



Variable	Test CER	Certificate type	Trained from model	Count training-data	Count test-data	Count inference-data
			almindelige dødsattester			

Table 7: Overview of training results and number of data points for training, testing and inference for read variables in HDAR.

The models that read the date of death, both for the regular death certificates and for the coroner's certificates, only read the day and month, and not the year. The reason for this was that for the coroner's certificates the year often does not appear on the certificate at all. Instead, the year of death appears from the metadata on the archive series, which indicates the storage location, which is the same as the year of death. For the regular death certificates, the year for the date of death is often not filled in either, and the quality of the reading was therefore not satisfactory. Instead, the year from the metadata for the date of death variable is used for both certificate types.


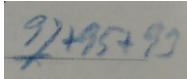
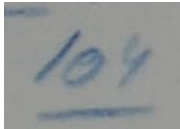

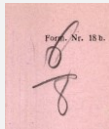
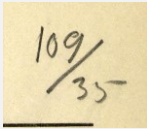
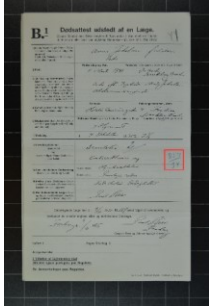
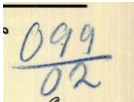

Inference is run with all models on the DB345 dataset. In addition, inference is run with the models for main disease and manner of death on the DB2 set, as these variables were not included in the DB2 set. For each image, the result of all models run with the associated probabilities, path to model checkpoint and timestamp for model run are saved in the associated .json file. For each variable that is read, a .csv file is also saved where the file path to the image and the top 5 TrOCR predictions with their associated probabilities are saved. These .csv files are used as input to a combined database for further work with HDAR.

## Cause of death code

Almost all death certificates have a handwritten cause of death code on the certificate itself. This code is written by the Danish Health Authority in order to code the cause of death from the certificate against the Danish Health Authority's official list of causes of death, each of which is represented by a code. The cause of death codes on the death certificates vary both in format and in location on the page throughout the period covered by the project and geographically depending on where in the country the certificate is from. Table 8 shows an overview of the different formats of cause of death codes, where they are used geographically and in which period. Please note that this is a rough overview and that there is overlap between the formats, so that some formats are found in several periods/geographical locations. The reading of these codes is important for the HDAR project, as this allows us to avoid having to interpret ourselves how different diseases have been understood and classified historically.

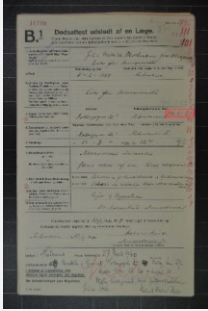
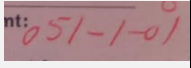
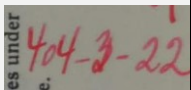
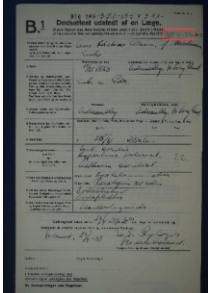
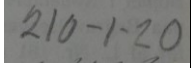
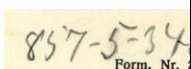
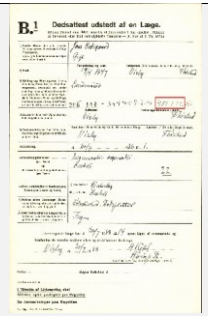
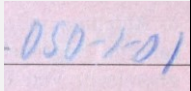
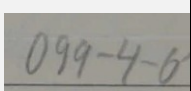




Format type	Description	Period	Location	Placement on certificate	Examples
0	Blue number in margin	1915-1930	Jutland and the islands		 
1	Division in upper right corner	1915-1936	Copenhagen		 
2	Blue division in margin	1931-1936	Jutland and the islands		 





Format type	Description	Period	Location	Placement on certificate	Examples
3	3digits-1 digit-2digits with red in margin	1939-1940	Entire country		 
4	3digits-1 digit-2digits in upper right corner	1938	Entire country		 
5	3digits-1 digit-2digits in margin with pencil or in blue	1937 (+ some from 1938)	Entire country		 





Format type	Description	Period	Location	Placement on certificate	Examples
6	Red "division" in margin	1941-1943	Entire country		
Ligsyns-mands-attest	Blue number in margin	1915-1931	Entire country		

Table 8: Overview and examples of different formats of cause of death codes that are manually written on death certificates by the Danish Health Authority. The format and location of the code on the page varies over time and depending on which geographical part of the country the certificate comes from.

For some formats, two different codes are given, e.g. for the bottom example of format type 6 and the top example of format 0 in table 8. Here, the top code is most likely the code for the main cause of death, while the other code(s) refer to secondary causes of death or secondary causes of death. In the HDAR project, only the top/first cause of death code has been read. The reading of the cause of death codes is standardized so that they always appear as a 3-digit number. If the code on the certificate consists of only 1 or 2 numbers, zeros are prefixed until the code is 3 digits.

### Segmentation and transcription

To identify and extract the cause of death code from all the death certificates, a YOLO model is trained for each format type. A dataset consisting of 1000 manually annotated images is used for training each model. From this, 50 images are set aside for validation.

After YOLO training, approximately 11,000 minipics (equal to minipics in the right column in Table 8) of cause of death codes are extracted to train an HTR model to read the codes. The 11,000 minipics are evenly distributed between the different format types. Of the



11,000 minipics, 500 are reserved for a dedicated test set. A TrOCR model from microsoft/trocr-large-handwritten is fine-tuned with the same training parameters and training strategy as for the other TrOCR models in the project. A character error rate (CER) of 0.65% is achieved on the test set.

After training both the YOLO and TrOCR models, a single pipeline for reading the cause of death codes can be put together. Since there is only one format type of cause of death code for the coroner's certificates <1931, these are run in a separate run and through a simpler pipeline. For the regular death certificates in the DB2 and DB345 set, the pipeline is defined as follows: Hver attest bliver kørt gennem samtlige 7 YOLO modeller. For hver YOLO model der giver et hit, køres der en TrOCR aflæsning.

- If there is only one YOLO model that gives a hit and thus only one reading, this is inserted together with the file path, probability and which YOLO model gave the hit, into a corresponding .csv file.
- If there is more than one YOLO model that gives a hit but all TrOCR readings are the same, the first reading is inserted together with the file path, probability and the number of the first YOLO model that gave a hit, into the aforementioned .csv file.
- If there are several YOLO models that give a hit and these give several different TrOCR readings, the file path, the different TrOCR readings separated by "|", the different probabilities separated by "|" and which YOLO models gave a hit separated by "|" are inserted into another .csv file specifically for manual inspection. The images in this .csv must then be manually reviewed as a later step in the HDAR project. For ligsynsmandsattesterne <1931 består pipelinet af en kørsel med YOLO modellen trænet på ligsynsmandsattester samt en aflæsning med TrOCR.

After inference by the pipeline, the cause of death code has been read for all death certificates and all coroner's certificates for the entire period. For those certificates where a cause of death code has not been identified, their image ID has been logged. The reason why a cause of death code has not been identified may be that there is actually no code written manually on the certificate, but it may also be that the code has been written somewhere on the page that the YOLO models have not been able to identify.

## Linking

The nomenclature for causes of death changes over time, and thus the link between cause of death and the associated cause of death code also changes. In order to interpret the read cause of death code on a certificate, it is therefore necessary to be in possession of the current linking table between cause and code for the period of the certificate. At the National Archives, we have the linking table for the period 1875-1930 and again from 1941-1951, but not for the period 1931-1940 inclusive. For all death certificates from the 1930s, it is therefore only possible to read the cause of death code but not to interpret which cause of death the code refers to. This is handled by manually comparing a number of certificates from each cause of death code and the cause of death nomenclature. As an alternative, the HDAR project has attempted to link the read causes of death from the 1930s to the current cause of death codes from 1941. The linked code should not be



understood as a replacement that should overwrite the read cause of death code, but instead as a supplement to interpreting the cause of death.

It has been shown that the Danish Health Authority's choice of cause of death code does not only depend on the cause of death itself but also on which diseases the deceased suffered from and which contributed to the death. For this reason, not only the reading of the cause of death field but also the reading of the main disease field is included in the project's linking between certificates from the 1930s and cause of death codes from 1941.

To perform this linking, a sequence classification model is trained whose architecture is named CANINE . The pre-trained model is loaded through the Transformers library and initiated from the checkpoint google/canine-s from Hugging Face. The combination of the reading of the fields with cause of death and main disease for all certificates with layout types A1 and B1 from 1941-1943 is used as training data for fine tuning. Only certificates with layout types A1 and B1 are used, as these are the only layout types where the field for main disease is included. For each certificate in the training set, the reading of cause of death and main disease is combined into one string and separated by a semicolon. The ground truth for training is the cause of death codes read by the project for 1941 and 1942. For 1943, there is an overlap between HDAR and the existing digital DAR. The ground truth for certificates from 1943 is therefore an official cause of death code retrieved from the digital DAR. In total, the training set for fine tuning the CANINE model consists of 102,698 pairs of a string in the form '[cause of death];[main disease]' and a 3-digit cause of death code. Of this, 10,000 data points are reserved for a dedicated test set. Of the remaining training data points, 20% are used for validation.

Training is performed with a cross entropy loss function and with the AdamW optimizer with a fast learning rate of  $1e-5$ . Training is performed with a batch size of 16. Accuracy is used as the evaluation metric, and training is performed for up to 100 epochs with early stopping and a patience of 20 epochs. After training, an accuracy of 89% is achieved on the test set. The fully trained model is named DeathCANINE40A1B1.

After training, inference is run on all A1 and B1 certificates from 1931-1940, which includes approx. 75,000 certificates from the DB2 set and approx. 279,000 certificates from the DB345 set. For each certificate, the file path, cause of death reading, main disease reading, DeathCANINE40A1B1 prediction and prediction probability are saved in a .csv file for the DB2 and DB345 sets, respectively.

The results of the CANINE models have not been included in the finished HDAR, as using the original cause of death codes were deemed better for all use cases.

### Coding occupation with HISCO

As an additional part of HDAR, all readings of occupational fields have been attempted to be linked to the official standardized HISCO (Historical International Standard Classification of Occupation) codes. This linking is done with the model OccCANINE<sup>12</sup>,

---

<sup>12</sup> <https://github.com/christianvedels/OccCANINE>



which is a CANINE model fine-tuned on 14 million pairs of job descriptions and HISCO codes, and is available on GitHub and Hugging Face. No further fine-tuning is done on the OccCANINE model, but only inference is run. Inference is run on readings of the occupational field for both the regular death certificates and the coroner's certificates <1931 for both the DB2 and DB345 sets. OccCANINE predictions for the 4 inference runs are saved in their own .csv file where the FilePath to the certificate, reading of the occupational field, HISCO code from OccCANINE and predictions probability are included.

The results of OccCANINE were not included in the finished HDAR, as it was deemed necessary to further clean up the results of the occupation-transcription before getting good OccCANINE predictions.

## Compute facilities

All classification models have been trained and inferred on the local, secure, offline computing facility Deep Thought.

All YOLO models for segmentation and for identifying cause of death codes have been trained and inferred on Deep Thought.

For the HTR models for name, birthday, date of death, occupation and cause of death for the regular death certificates, the training has taken place in two steps. The first step has consisted of an initial fine-tuning on the cloud-based HPC computing facility UCloud<sup>13</sup>, which is offered by SDU and ISO 27001 certified. The fine-tuning on UCloud has consisted of training data from layout types B, B1, B2 and E, which deal with deceased persons over 1 year old. The remaining layout types deal with deceased persons under 1 year old or stillborns, and are currently assessed based on a precautionary principle as sensitive personal data, which therefore may not be uploaded to UCloud. In order for the models to learn to predict on all layout types, models from UCloud have subsequently been further fine-tuned as step 2 on the part of the training sets that contain certificates with layout types A, A1, A2, C and D. By using UCloud for initial fine-tuning, the training time for the relevant models has been significantly reduced. This is because the use of UCloud makes it possible to run multi-GPU training, whereby the training is spread across multiple GPUs, whereas Deep Thought only has a single GPU.

For the remaining HTR models for the regular death certificates, and for all HTR models for the coroner's certificates <1931, all training has taken place on the local Deep Thought facility. All inference for the HTR models has taken place on Deep Thought.

Training and inference of the DeathCANINE40A1B1 model has taken place on Deep Thought, just as inference of OccCANINE has taken place on Deep Thought.

In addition to the initial fine-tuning of the 5 HTR models mentioned, and during the early development of a strategy for segmentation, computational resources have also been used on UCloud on experiments with models that did not end up being used in the project and which are therefore not further described here. In total, 1610 GPU-hours have been

---

<sup>13</sup> <https://cloud.sdu.dk>



used on UCloud in the HDAR project at 19 DKK per unit and 7611 core-hours at a price of 2.74 DKK for 32 core-hours. This gives a total of approx. 31,000 DKK.

