# Feature Engineering for Neural Networks

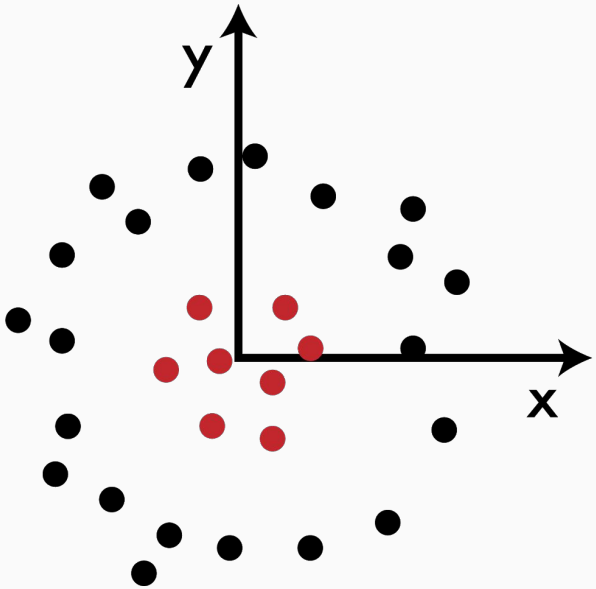
Benjamin Kiessling

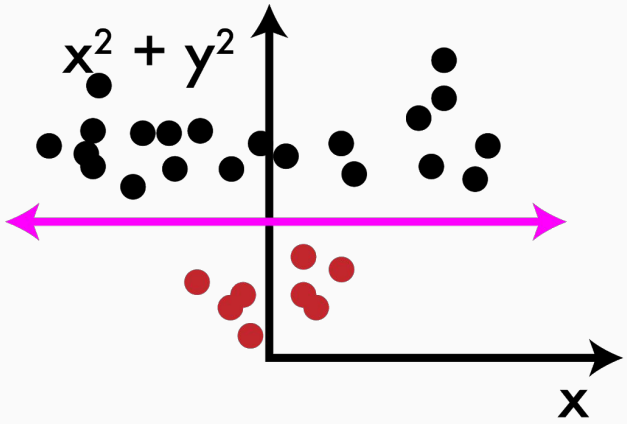
October 4, 2023

# What is feature engineering?

Arguably the core problem of machine learning (especially in practice).

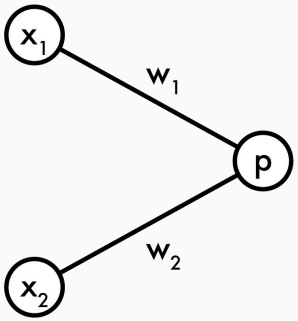
ML models work well if there is a clear relationship between the inputs and outputs of the function you are trying to model.



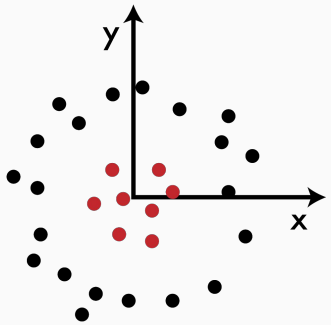



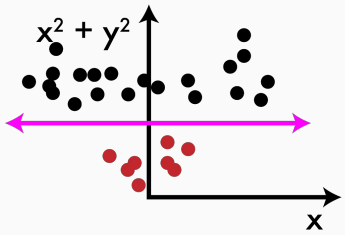
# What is feature engineering?

Linear models cannot learn transformations of features, only use existing features. Humans must create good features.




$$p = f(w_1x_1 + w_2x_2)$$

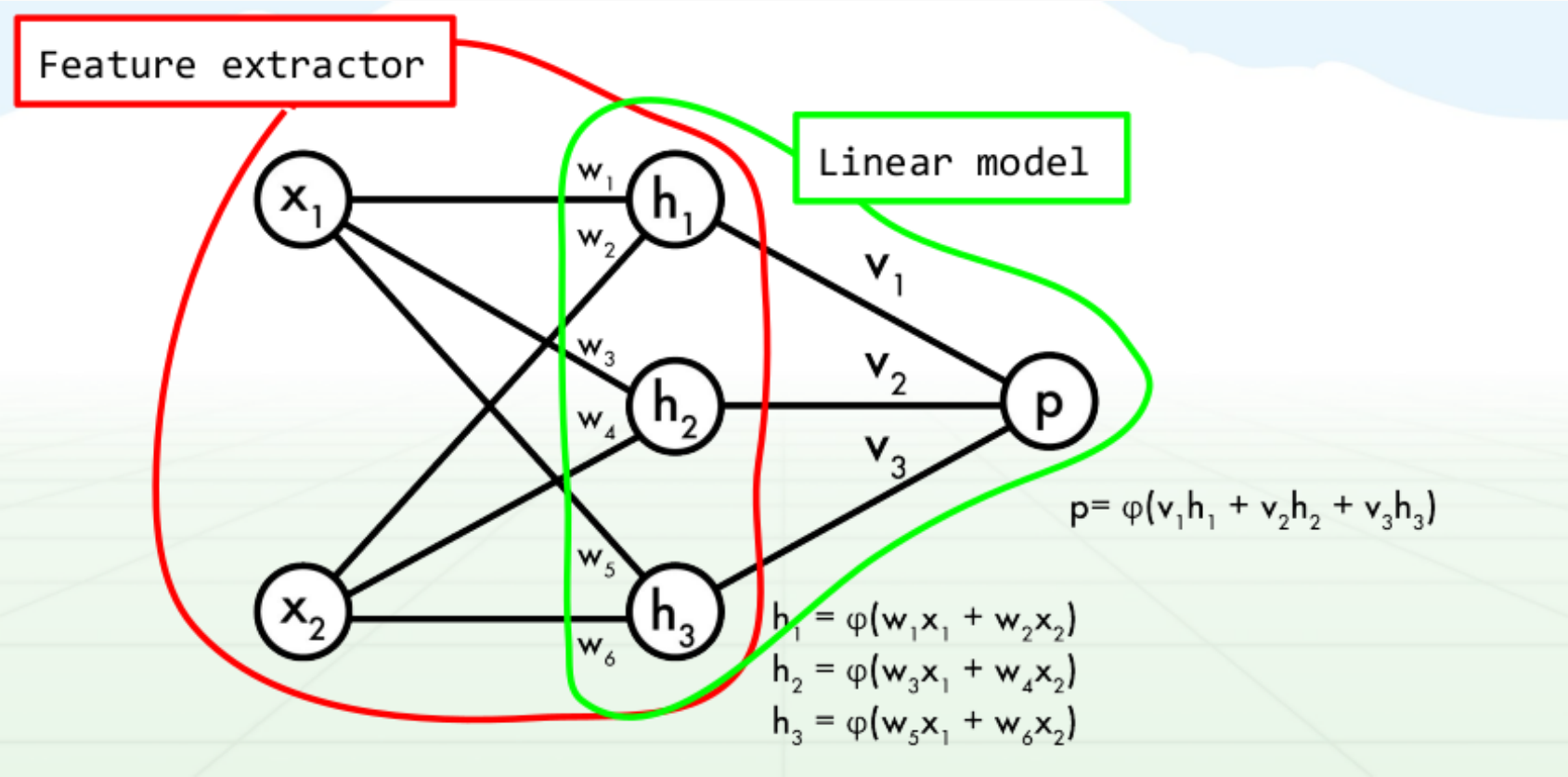


# What is feature engineering?

Neural networks can, in principle, learn arbitrary transformations of input data.

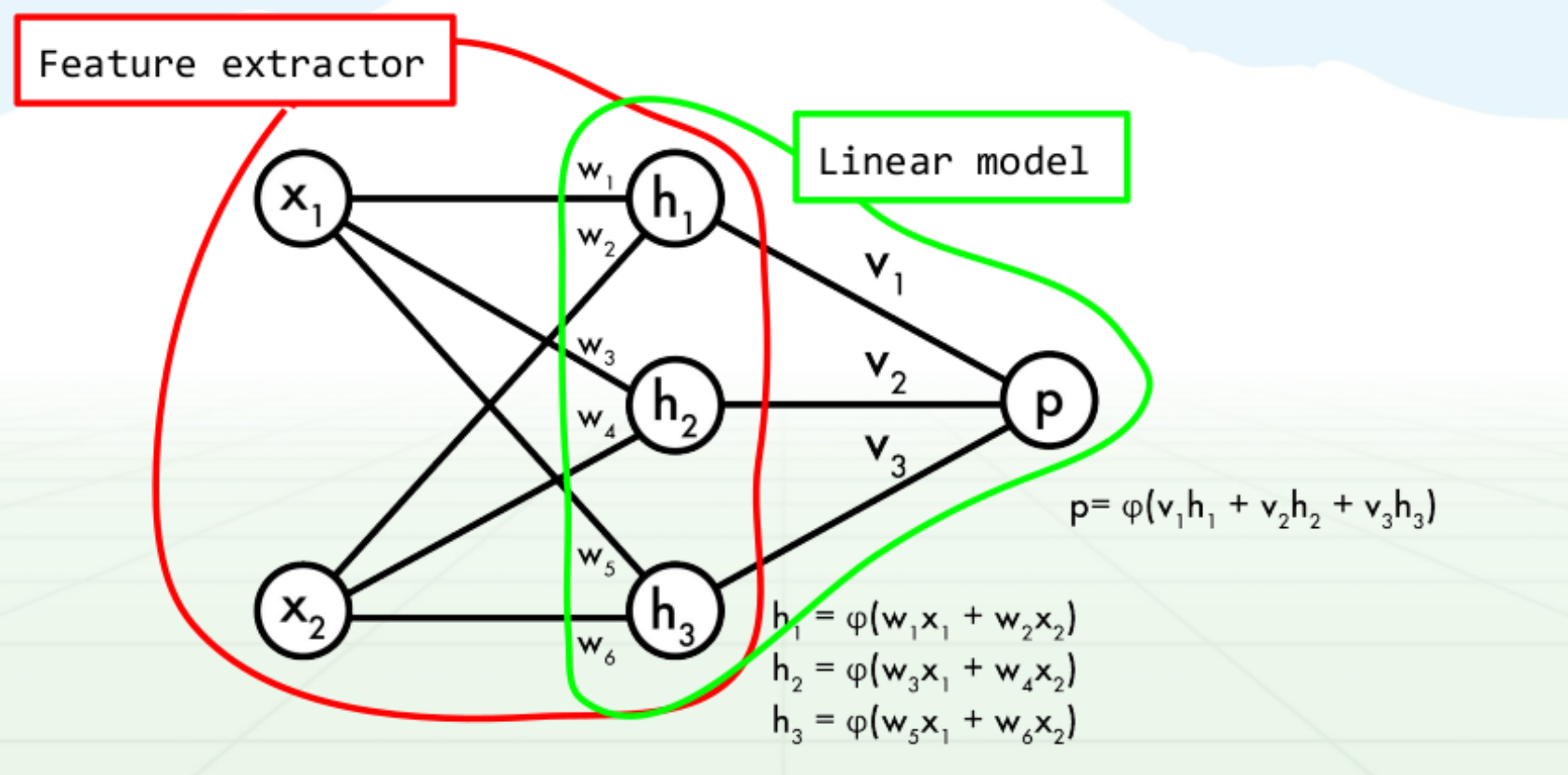
They are universal approximators, i.e. sufficiently deep networks can learn any mapping between inputs and outputs.



# What is feature engineering?

In practical terms some mappings are easier to learn than others.

Learning abstract representations requires more sophisticated networks, more training data, more experimentation, ...



# What is feature engineering?

It is often advantageous to utilize simpler methods with some kind of postprocessing if the data basis is slim or computational ressources are limited.

Likewise it is often important to factor in the ease of obtaining/annotating training data.

Feature engineering when using deep neural networks is finding the appropriate data model for the input-output mapping given the constraints of the method and data available.

# Example - Layout Analysis

Document layout analysis is a processing step in Optical Character Recognition (OCR).

Depending on the configuration of the OCR system the segmenter, the process performing layout analysis, has to find single characters, words, or lines on a page image that are then fed into a recognizer which converts them into text.

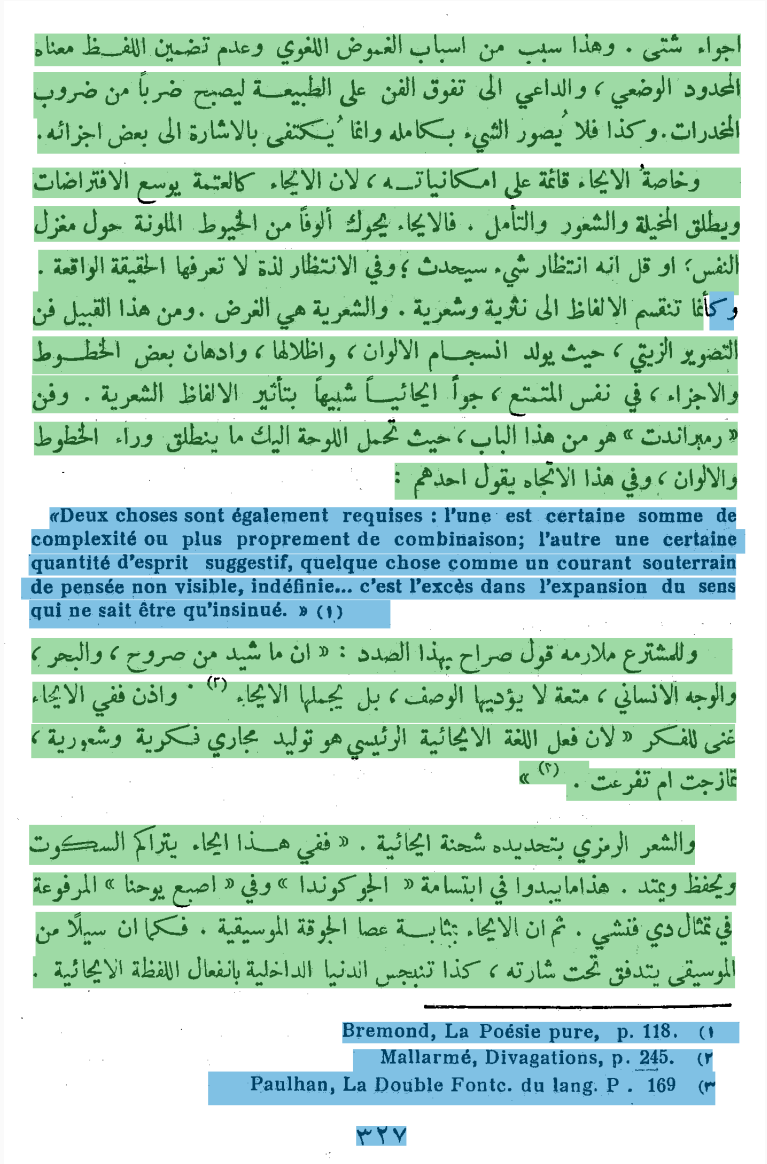
Modern text recognition systems operate on a line level, i.e. they convert whole lines at a time into text.

# Examples - Layout Analysis

Therefore one has to decide how to model individual lines in a way that they can be processed easily by the recognizer and how to model them for easy learning by a neural network.

# Examples - Layout Analysis

We can model lines as bounding boxes. They are easily processed by the recognizer.



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

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

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

وللشاعر ملارمه قول صراح بهذا الصدد : « ان ما شيد من صروح ، والبحر ،
والوجه الانساني ، متعة لا يؤديها الوصف ، بل يجملها الايحاء (٢) . واذن ففي الايحاء
غنى للفكر « لان فعل اللغة الايحائية الرئيسي هو توليد مجاري فكرية وشعورية ،
تمازجت ام تفرعت . » (٣)

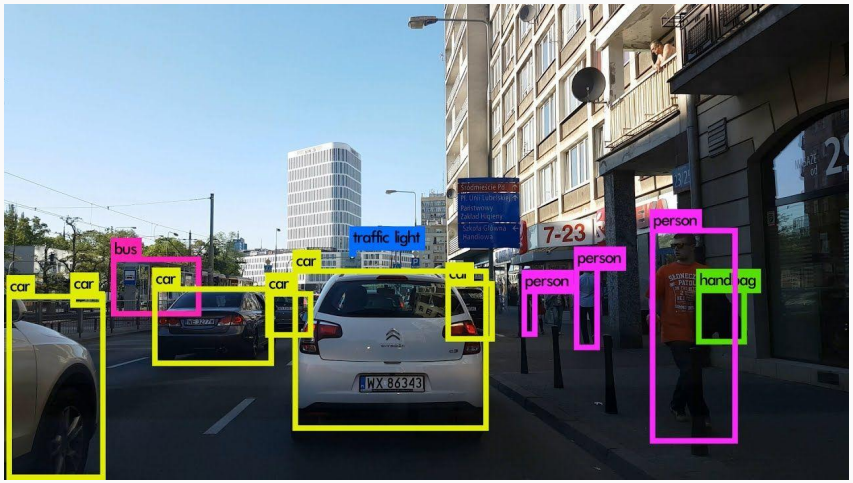
والشعر الرمزي بتحديده شحنة ايحائية . « ففي هذا الايحاء يتراكم السكوت
ويحفظ ويمتد . هذا مايبدوا في ابتسامة « الجوكوندا » وفي « اصبع يوحنا » المرفوعة
في تمثال دي فنشي . ثم ان الايحاء بمثابة عصا الجوقة الموسيقية . فكما ان سيلاً من
الموسيقى يتدفق تحت شارته ، كذا تنبجس الدنيا الداخلية بانفعال اللفظة الايحائية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

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

٣٢٧

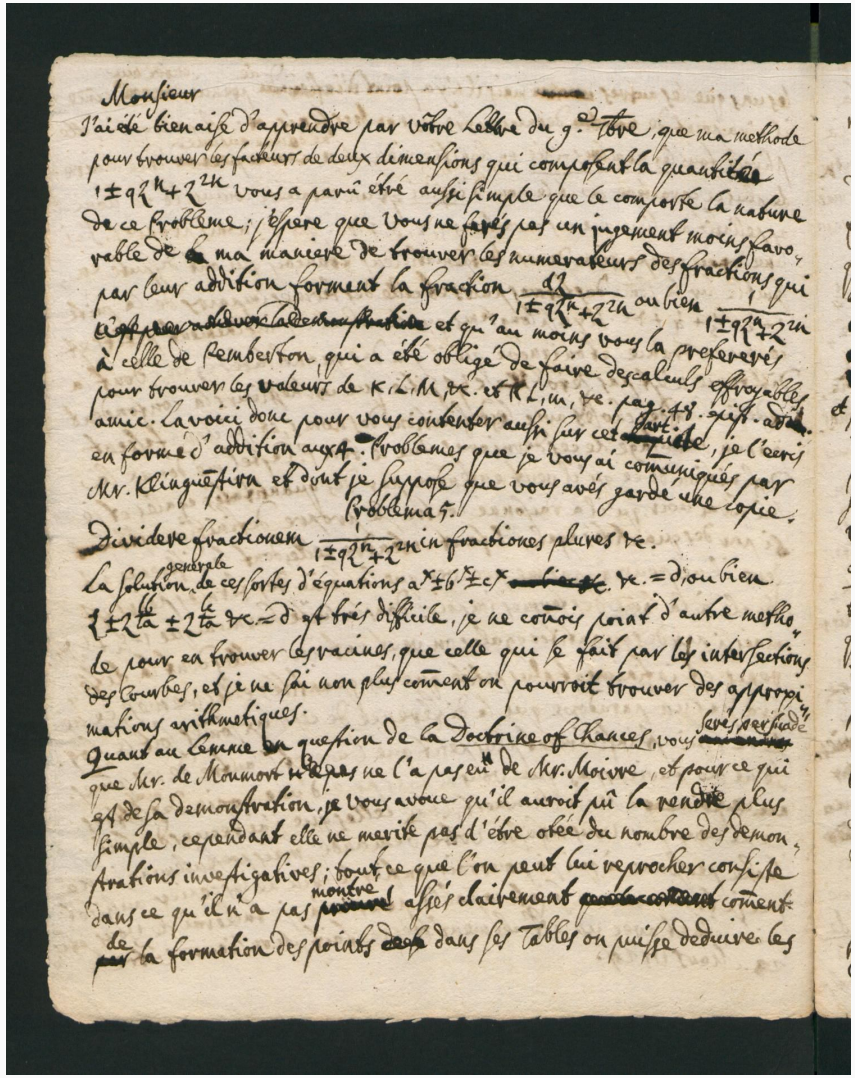
# Examples - Layout Analysis

And we've got methods that can find abstract boxes of objects. But they require a lot of training data (PASCAL VOC: ~8000 images).



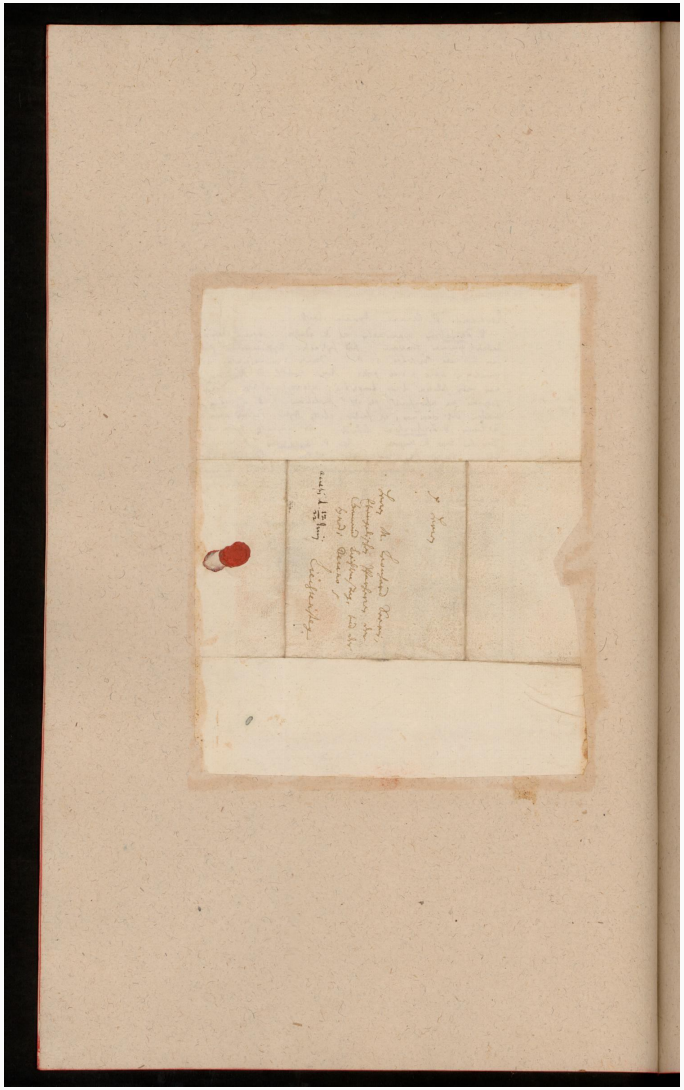
# Examples - Layout Analysis

In addition, we might want to apply our OCR pipeline on manuscripts. Manuscript lines might not be straight, so drawing boxes around them will be difficult.



# Examples - Layout Analysis

They might also be rotated.

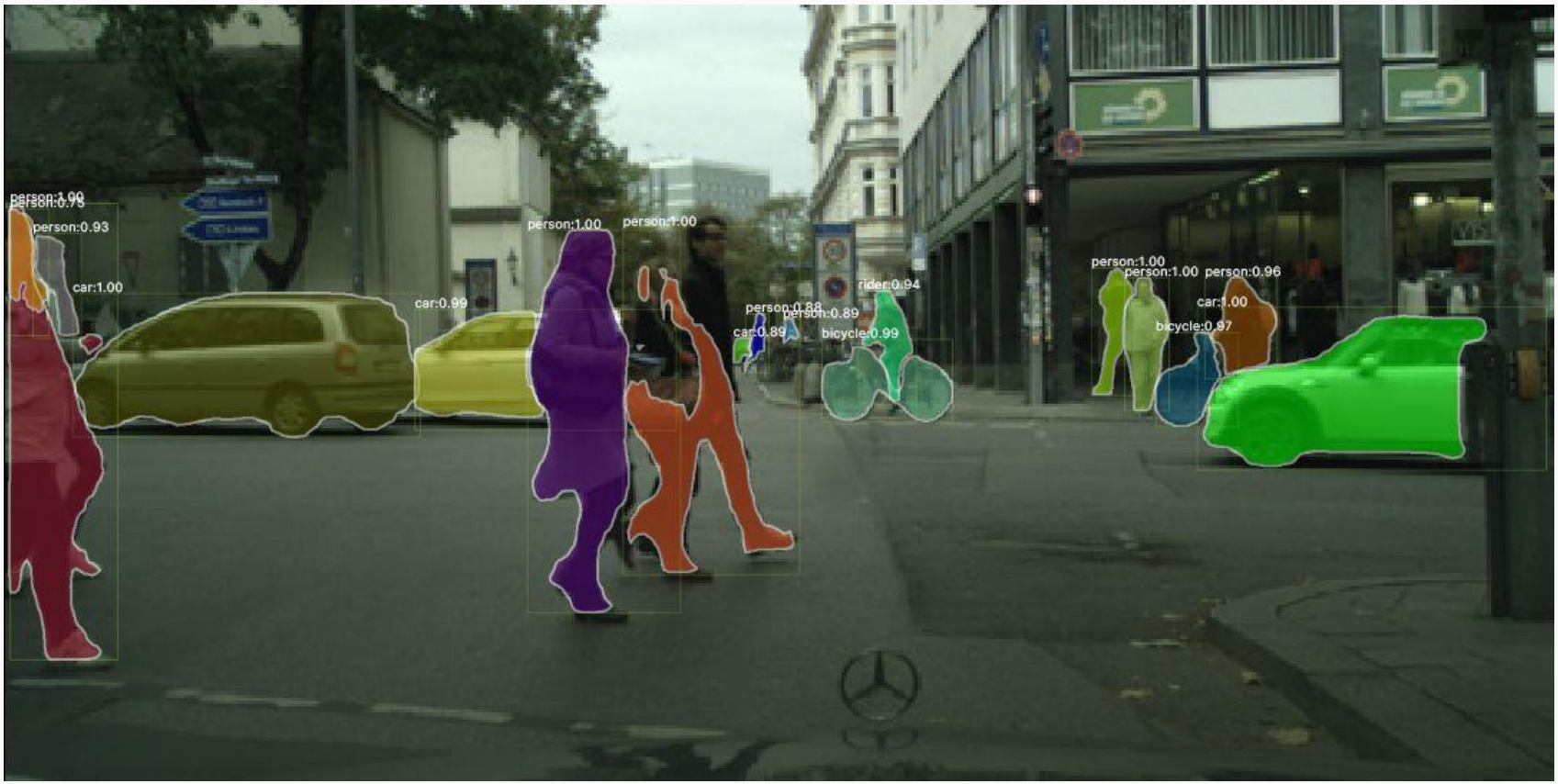


# Examples - Layout Analysis

Or both!

# Examples - Layout Analysis

Instead of object detection we could switch to instance segmentation methods.

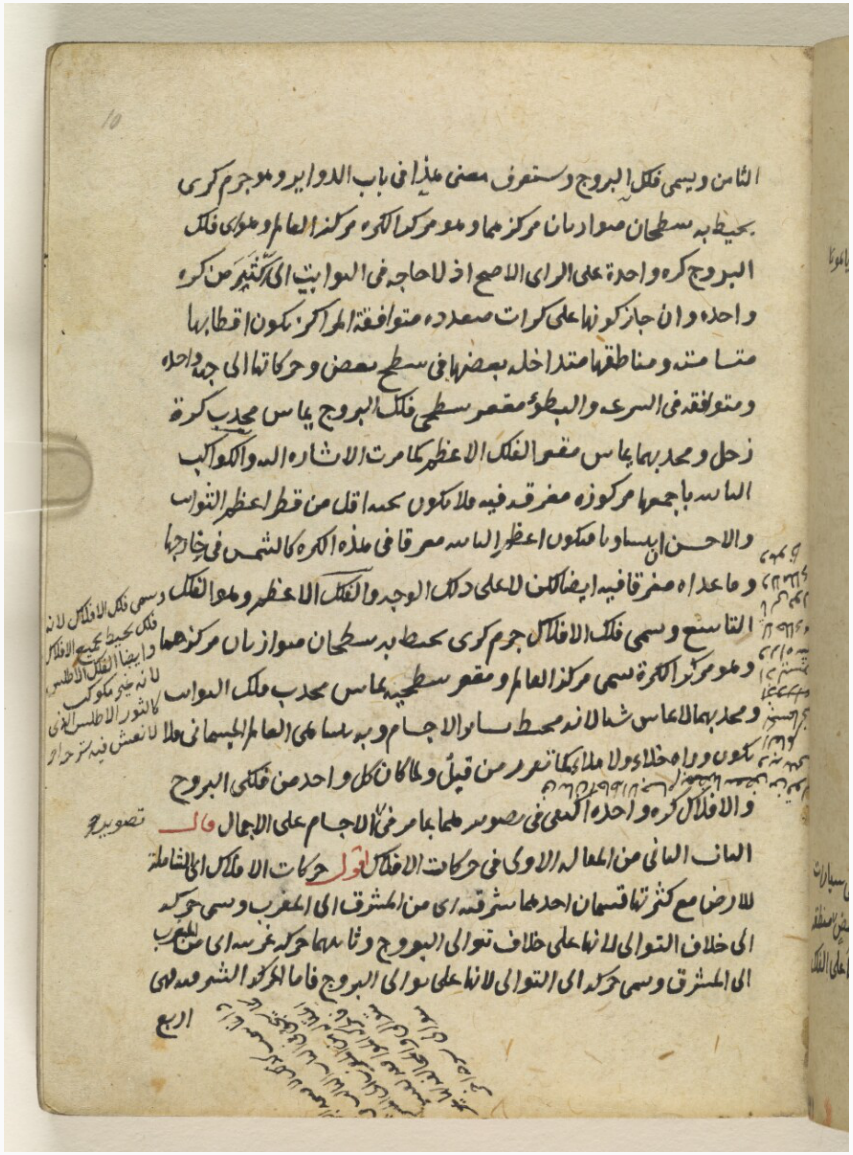


# Examples - Layout Analysis

Instead of object detection we could switch to instance segmentation methods.
But these are even more demanding than object detection methods.
We need more training data and annotating polygons is even more complicated that boxes.

# Examples - Layout Analysis

Further, our recognizer doesn't deal as well with curved lines that aren't rectified/rotated.
It completely fails on lines written on their head as well.

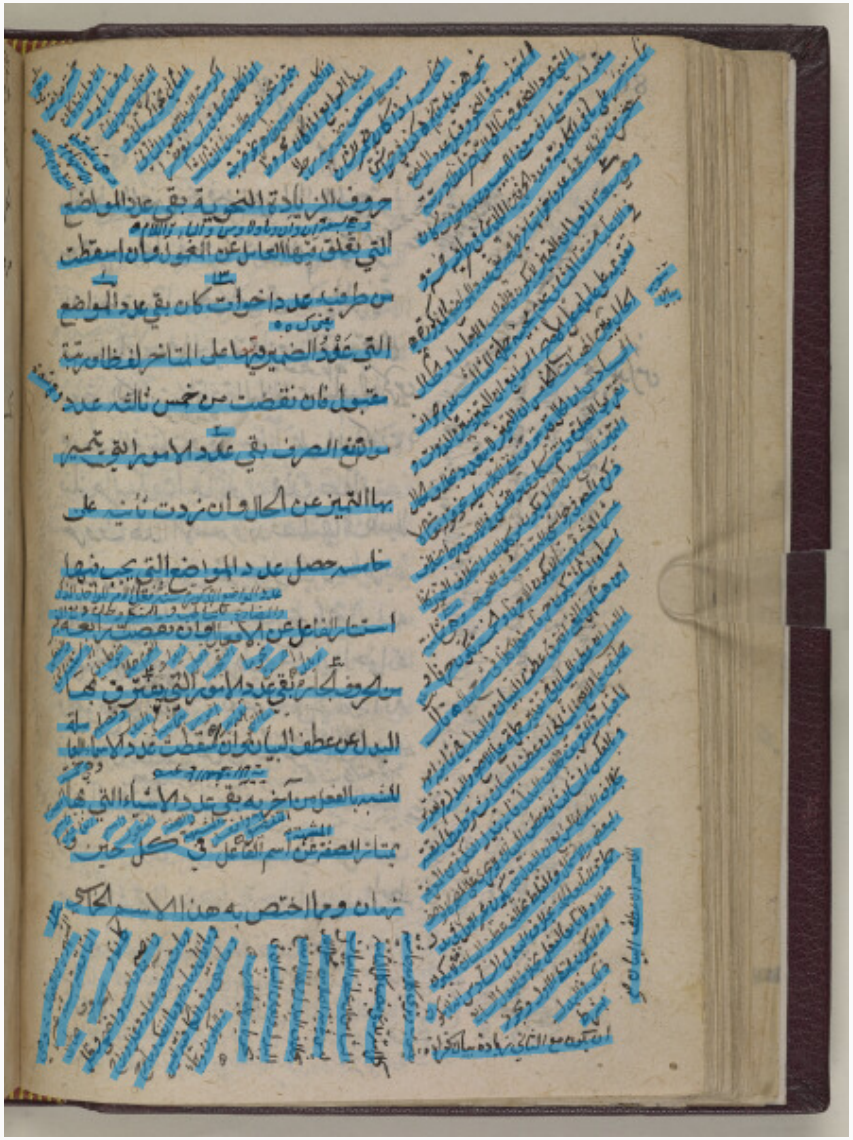


# Examples - Layout Analysis

Our system should therefore be able to model arbitrarily shaped lines, rectify, and orient them.
They should be easily learned.
Training data should be quick to produce.

# Layout analysis - Baselines

We can split the line representation into multiple parts to ease processing.
A baseline and a bounding polygon.
The baseline is defined as the virtual line upon which most characters rest (or hang from in the case of Hebrew).
Baselines aid in rectification as we can map a curved line easily onto a straight line.



# Layout Analysis - Baselines

Baselines can also easily be oriented by defining a start and end point.

# Layout Analysis - Bounding Polygons

The bounding polygon contains all the textual content for a baseline.
By drawing the polygon we can suppress image data not belonging to the line.
Combining both structures into a data model we achieve rectifiability, orientability, and arbitrary shapes.
Lines in this model are as easy to process by the recognizer as box lines.

# Examples - Baseline Recognition

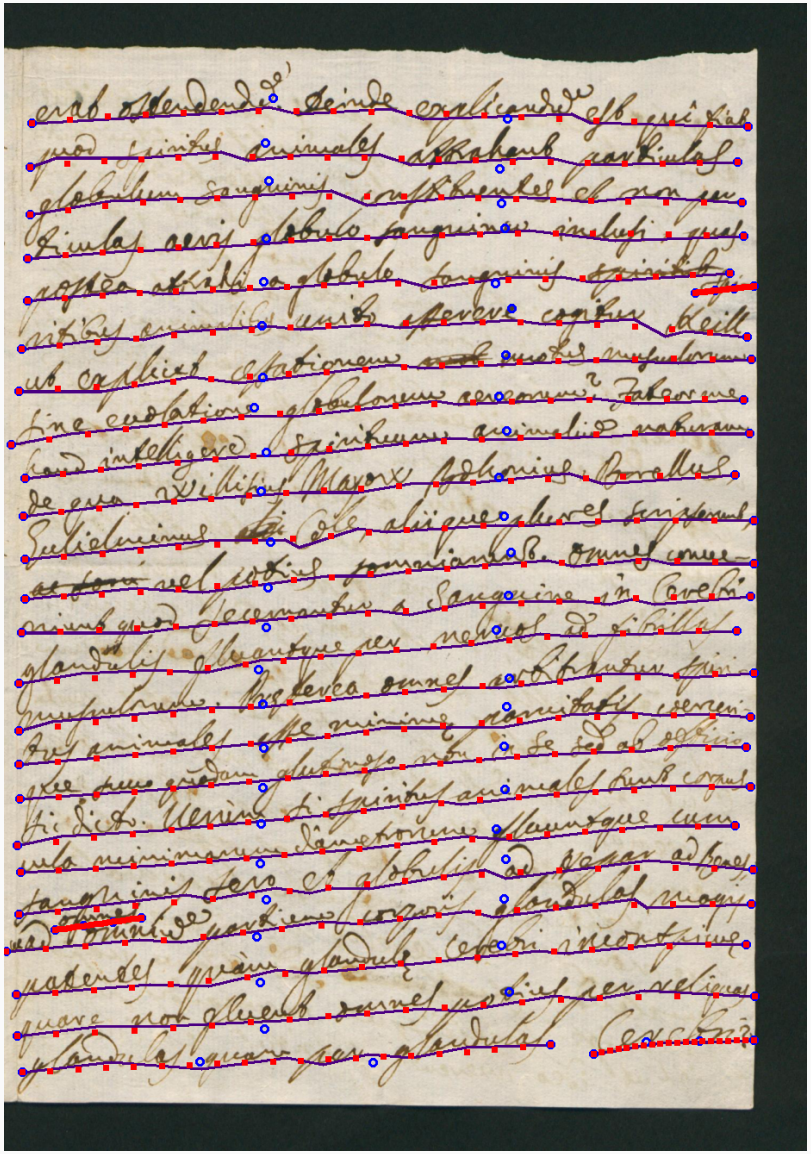
The question remains how we can build a segmenter which outputs this data representation.

Some instance segmentation methods (TensorMask, ESE-Seg) do not produce instance-wise semantic segmentation masks but abstract shape representations.

We could theoretically build a segmenter that predicts abstract baseline and bounding polygon representations.

# Examples - Baseline Recognition

A few object detection methods (DETR) can also be adapted to produce non-box output (CurT).



# Examples - Baseline Recognition

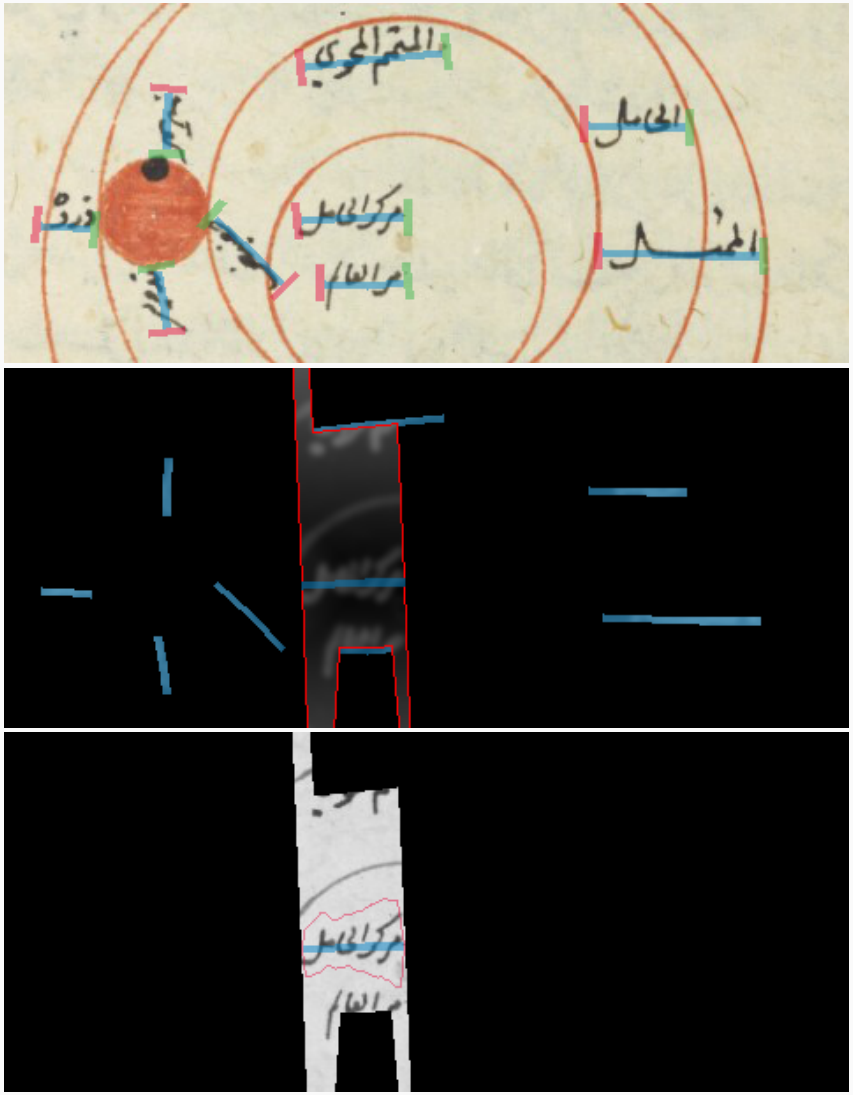
These methods are very slow and again require lots of training data.

While one part of our data model is very quick to annotate (baselines) the other part (bounding polygons) takes a lot of time.

If we can find some way not having to learn the bounding polygon we can create a lot more training data quicker and our model becomes simpler!

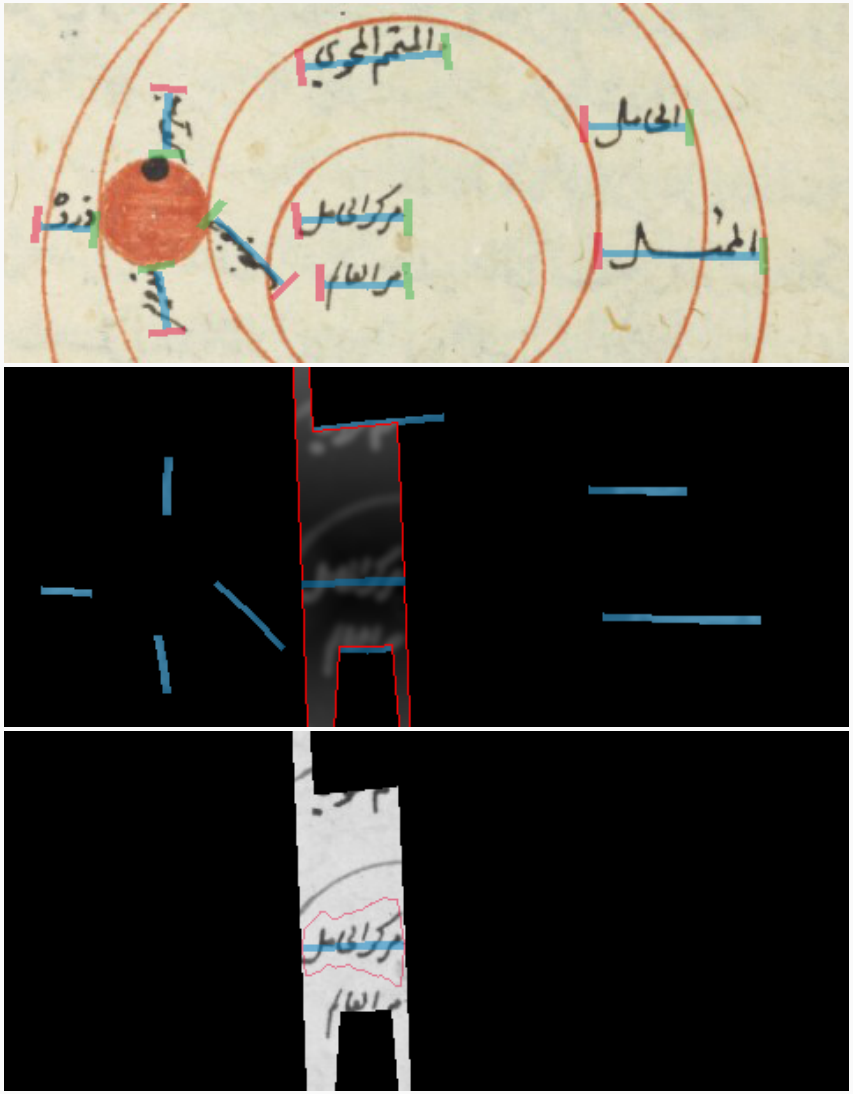
# Combining CV and ML

There are multiple ways we could compute a bounding polygon using low-/mid-level computer vision methods.
A fairly robust one is seam-carving.
Seam carving finds a path through an image that has the lowest cumulative energy, i.e. the lowest sum of color change.



# Combining CV and ML

Seam-carving the image between our learned baseline output, we can compute bounding polygons reliably.
The segmenter will use both traditional computer vision and machine learning, forming a hybrid system.
Such hybrid systems are very common.

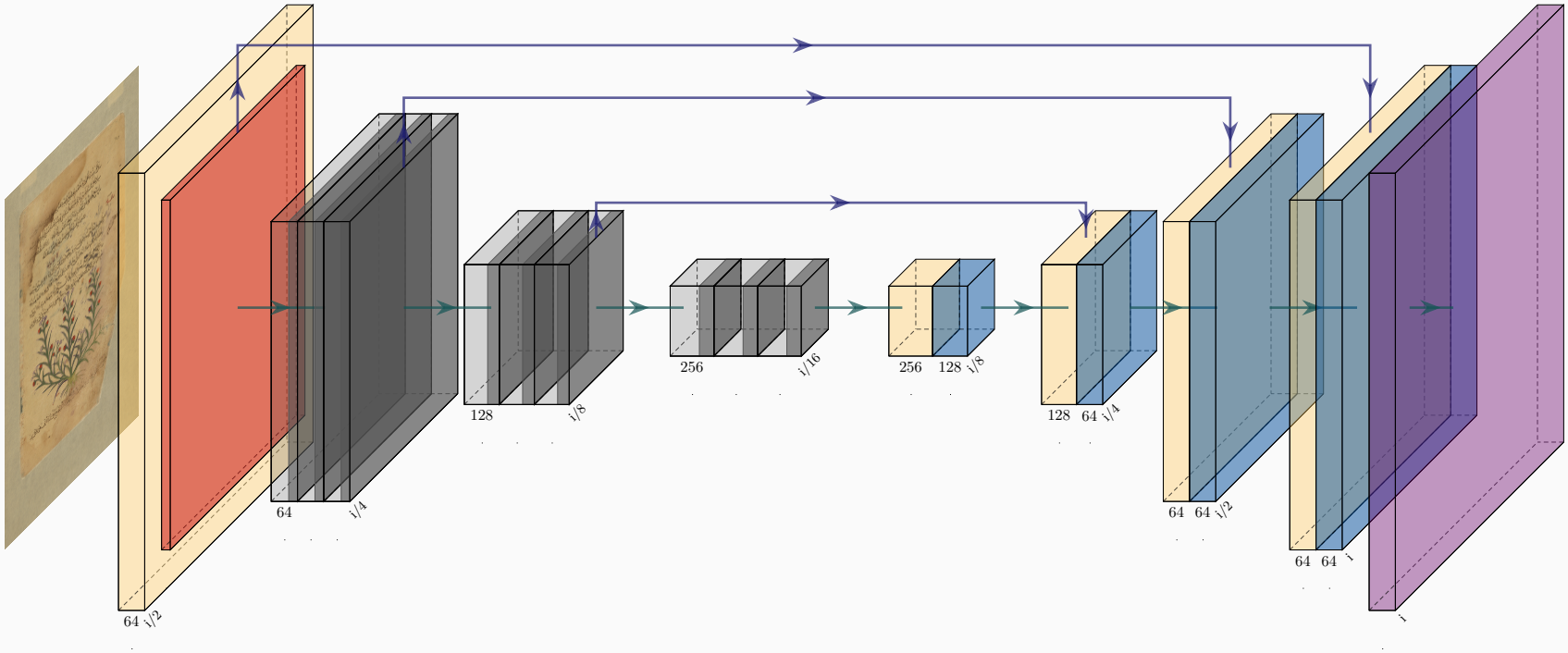


# Examples - Baseline Recognition

We have simplified the learned part of our data model to oriented baselines.
Instead of object detection we can use semantic segmentation to label pixels as belonging to a baseline.
This works well as baselines are not overlapping, except in rare cases like palimpsests.

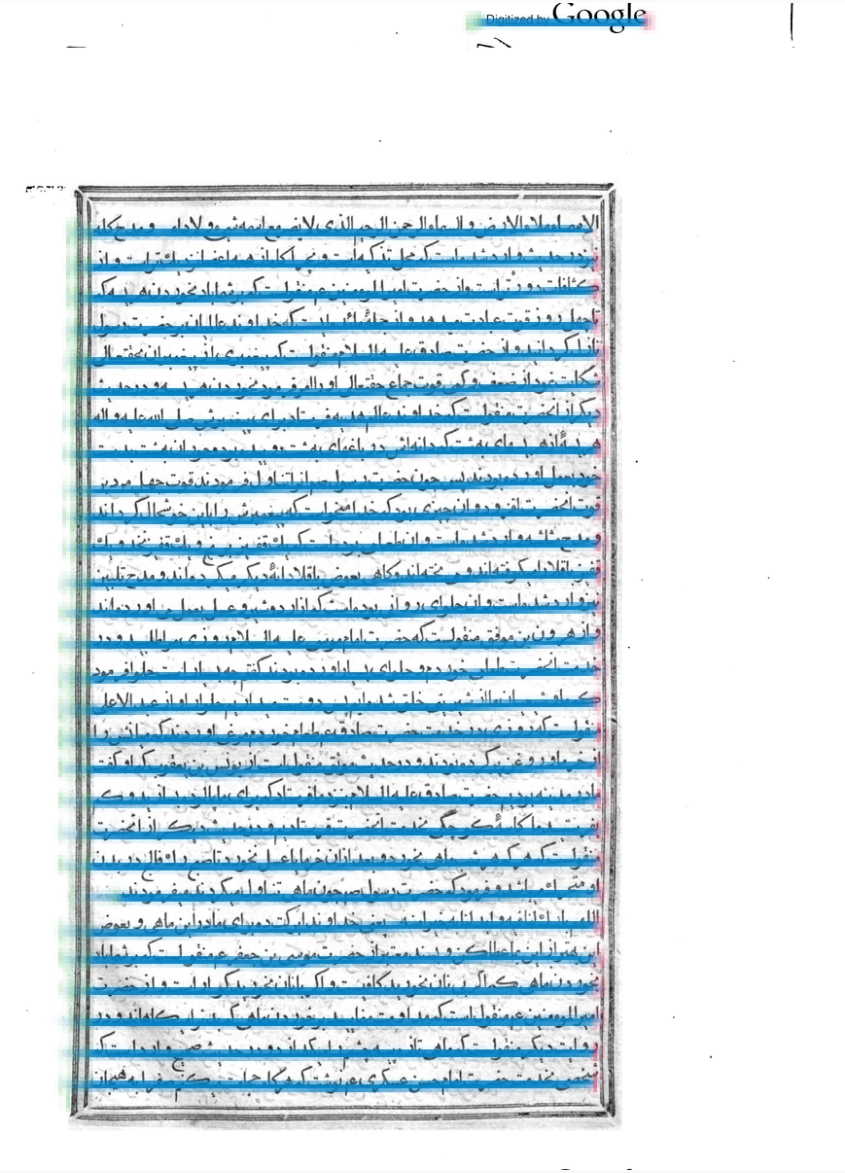
# Baseline Recognition with Semantic Segmentation

A simple U-Net is sufficient for our purposes and can be trained well with a low number of samples (~50 pages).



# Baseline Recognition with Semantic Segmentation

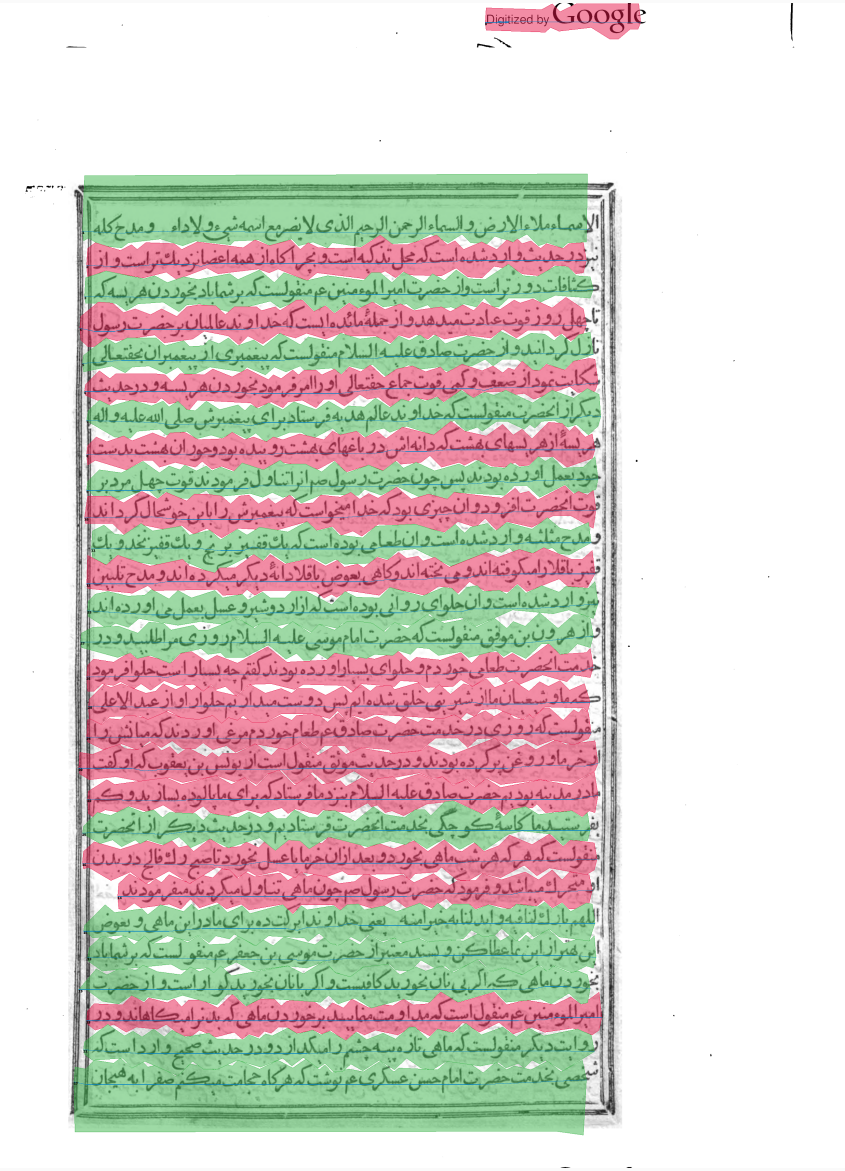
The output of our network classifies each pixel into belonging to either a baseline, a line start, or a line end. With some minor postprocessing we can compute an abstract, oriented baseline out of the classification map.



# Baseline Recognition with Semantic Segmentation

With some minor postprocessing we can compute an abstract, oriented baseline out of the classification map.

We then run our traditional computer vision seamcarve with the learned baselines and retrieve both parts of our data model.



# Synopsis

When we try to solve a problem with computer vision and machine learning deciding on a data model for the outputs (and inputs!) is the most important decision.

Apart from considerations of ease of learning, the effort of creating training data should also be taken into account.

In general: abstract representations are harder to learn than simple transformations.

# Synopsis

With complex data models we almost always construct a pipeline of different methods.

This works especially well if we can easily transform/create a desired abstract representation with low-level computer vision methods that would be hard to learn for a neural network.

It also allows the use of off-the-shelf algorithms.