DSP: Fire Risk Profiles

Creating structured data from floor plan images

Anton Kozačkov, Elisabeth Kräman, Jelle van Elburg, Marta Turek & Will Chien
Supervisor: Michael Kern
University of Amsterdam
In collaboration with:
Municipality of Amsterdam
Amsterdam, The Netherlands

ABSTRACT

Fire safety inspectors from the Amsterdam fire department are responsible for conducting annual inspections of commercial structures to ensure that buildings comply with fire safety codes. Due to resource constraints, not all buildings can be inspected annually and a subset of properties must be prioritised for inspection. A model to determine property-level fire risk has been developed to provide a data-driven process for property prioritisation, however, the model requires additional features to improve its output. In this report, an approach is proposed to extract structured feature data from unstructured building floor plan images by using a Convolutional Neural Network. The model is able to recognise elements in floor plan layouts and is trained to complete two tasks: to predict room-boundary elements such as doors and windows and to predict room functions. These data can then be extracted in structured form and be used as additional input features for the fire-risk model.

KEYWORDS

floor plan, boundary detection, room function, system design, convolutional neural network

1 INTRODUCTION

In the Netherlands 71 fires with over a million euros in damages were reported in 2018 [1]. The overall fire damage per year often exceeds half a billion euros. Nearly half of these fires occurred in commercial buildings. Municipal fire departments, such as the Amsterdam fire department (*Brandweer Amsterdam-Amstelland*), are responsible for enforcing fire codes to reduce the risk of structure fires. Fire safety inspectors from the fire department conduct annual inspections of various types of non-residential properties, such as commercial, industrial and governmental structures to ensure that these buildings comply with decrees such as the 2012 Dutch Building Decree (*Bouwbesluit*) [2]. This decree includes statutory rules and regulations to improve fire safety. The purpose of these inspections is to detect regulatory violations that, if undetected, may result in a fire incident.

At the same time, major cities in the Netherlands, are expanding rapidly. Amsterdam expects a growth in population of about 20% by 2035 [3]. This growth in population will be accompanied by an increase in both commercial and residential buildings. In 2019 alone over 5 000 new buildings were constructed in Amsterdam, which constitutes an increase of more than 1% in the total number of buildings[4]. However, many municipal fire departments, including the Amsterdam fire department, are unable to inspect all the commercial properties in the city annually. Therefore, in a given year they must determine which subset of properties warrant inspection. To inform this decision, many fire departments rely on some form of rule-based heuristics for determining which commercial properties to inspect. This kind of approach to determining inspection priority does not involve a risk-based evaluation of the fire risk at the properties selected.

To address this weakness in fire inspection prioritisation, the data science team at the Municipality of Amsterdam has developed a model to determine the likelihood that a fire incident will occur in a commercial property. The risk assessment for a building is based on three measures. First, the chance of a fire incident occurring in a building. Second, the probability that a building or the usage of a building is not up to standards and regulations. Third, the effect that a potential incident would have. The goal is to provide fire safety inspectors with a risk-based data-driven process for the prioritisation of commercial properties to inspect.

To improve the insights the model provides into the fire risks of commercial buildings in Amsterdam, additional data inputs are required. Specifically, features need to be constructed as inputs for the model from historical, real-time or unstructured datasets. However, easily attainable input features are not readily available, which hinders model expansion and makes feature engineering a key challenge of this project.

To model the economic cost of a fire incident or to estimate infrastructure damage, a number of inputs are required. These inputs include, but are not limited to, building layout features such as floor dimensions, number of compartments, size and layout of compartments, construction materials, compartment elements such as the number of doors and windows, interior and exterior wall lengths etc. [5]. The layout of most commercial buildings is represented in floor plan images and these images contain within them valuable information about building characteristics in unstructured data form. Extracting this unstructured data from floor plan images to structured data using image recognition techniques would yield a number of potentially useful input features that can be added to the fire-risk model.

In this report an approach for creating structured data from floor plan images by using a Convolutional Neural Network (CNN) is proposed. Various fire risk assessment methods are outlined in section 2 and the model leveraged for floor plan recognition is described in detail in section 3, with a discussion of implementation architecture, hyper-parameter tuning and model evaluation. Additionally, a web tool for converting images to data and representing that data is constructed and evaluated. First, the interaction design of the

system and the evaluation of the application prototype is discussed in section 4, followed by the technical details of the system design in section 5 and a final evaluation of the user interface is covered in section 6. Finally, in section 7 the implications of the findings and the limitations of the work are discussed.

2 RELATED WORK

2.1 Fire risk modeling

A considerable amount of literature is available on possible methods of fire risk assessment. Hadjisophocleous et al. [5] outline a variety of different quantitative measures of fire risk that can be used to inform a prioritisation model for fire inspection.

2.1.1 Model factors. Any effort to model the risk of a fire needs to consider both the factors of likelihood and the possible consequences of a fire event. According to Kaiser [6] per the Gretener Method, the risk is seen as the product of probability and potential impact of a fire event. As such, information is needed on both factors associated with the probability of a fire event occurring and the multitude of adverse consequences of a fire event. Without this information, a model cannot effectively determine the balance of these risk components and will be unrepresentative. The first stage of any fire risk model therefore is determining the factors impacting risk and the damage potential of a fire incident. Data on qualities of the building, such as room types and features, is a valuable input into any such model as properties of the building determine both severity and likelihood of fire incidents. For instance, doors and window provide egress routes and enable occupants to leave a building in case of a structure fire.

2.2 Floor plan image recognition

The extraction of building features is a subclass of computer vision tasks related to document segmentation and parsing, and the task of recognising elements or recovering information from floor plan images is an actively researched problem in the computer vision field. The goals of such models range from simple digitisation to recovery of high level information, such as area, layout and room function, from these images.

2.2.1 Algorithmic models. Early approaches followed on algorithmic extraction of lines and shapes from the images by means of applying filters to separate the walls and contours of the building from notation or auxiliary lines in the image, such as in the pipeline proposed by Ahmed et al. [7].This method however could not accommodate differences in notation style and, for example, relying on walls as always being the thickest line in the image.

2.2.2 Machine learning. Machine learning models could adapt to stylistic differences and additional complexity based on ground-truth labeling that could be expanded upon. The use of Convolutional Neural Networks (CNNs) to extract features from images has also become the norm in most computer vision tasks. Combining this Liu et al. [8] created a pixel-based prediction model using a CNN to detect junctions, such as corners and wall intersections and reconstruct the walls on a geometric basis by connecting these junctions. This approach had high average precision and recall, yet could not infer information other than wall layout and was furthermore trained on a dataset that was not diverse enough to generalise easily.

2.2.3 Extracting information. Methods focusing on extracting more high level information, such as room shapes, also utilised forms of machine learning. Dodge et al. [9] constructed a robust model that uses a Fully Convolutional Neural Network (FCNN) combined with optical character recognition to infer properties like room sizes and detect recognisable objects like washrooms and doors. Most recently, Zeng et al. [10] developed the DeepFloorplan Model an approach to use Deep Learning and effective post-processing methods to accurately recognise compartments, doors, windows and infer room function. Generally the success of these models is predicated on the choice of model and the availability of a labeled dataset, the construction of which is a major hurdle in the research. Since these datasets are usually sourced and annotated by the research team from a single source the homogeneity of the training data limits the accuracy and general applicability of a pre-trained model to more varied data.

3 MODEL

The DeepFloorplan Model [10] aims to recognise boundary floor plan elements such as walls, doors, windows and room regions. Additionally, the model is targeted to be able to handle rooms of non-rectangular shapes and walls of non-uniform thickness. Finally, the model recognises room functions in floor plans.

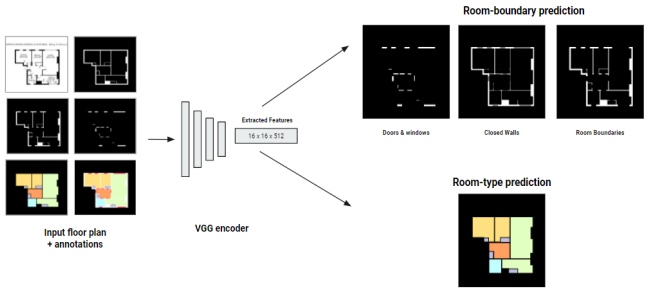

Figure 1: Model Architecture: Floor Plan Input via VGG Encoder and Decoded Room-Boundary and Room-Type Predictions

3.1 Network Architecture

Overall architecture. Figure 1 represents the overall model architecture. The model uses the floor plan images and supporting annotations as an input. The pre-trained weights of a VGG network [11] are used and via transfer learning the weights are applied from the VGG classification network (on ImageNet) to the annotated floor plan dataset. For each image, the pre-trained VGG network is looking at the annotations and updates itself to improve the classification of the input features. There are two main tasks in the network: one for predicting the room-boundary pixels and the other for predicting room-type pixels. Specifically, the network first learns the shared feature, which is common to both tasks, then it makes use of two separate VGG decoders to first segment images,

then deconvolve them and predict a label for each segment from a known list of labels. There are two separate label types that the network is trying to learn. First to predict room boundaries, including doors and windows, and second to predict room-types.

Cross-and-within-task weighted loss. Both of the specific tasks in the network involve multiple labels for various room-boundary and room-type elements. Given that the number of pixels varies for different elements, their contributions within each task need to be balanced. Additionally, there is a larger number of room-type pixels than room-boundary pixels, as such the contributions of the two tasks need to be balanced. To achieve this, a *cross-and-within-task weighted task* is designed in order to balance between the two tasks as well as among the floor plan elements within each task.

• *Within-task weighted loss.* The within-task weighted loss is defined in an entropy style calculation and defined as

$$L_{task} = w_i \sum_{i=1}^{C} -y_i log p_i, \tag{1}$$

where y_i is the label of the i-th floor plan element in the floor plan and C is the number of floor plan elements in the task; p_i is the prediction label of the pixels for the i-th element ($p_i \in [0, 1]$); and w_i is defined as:

$$w_i = \frac{\hat{N} - \hat{N}_i}{\sum_{j=1}^{C}(\hat{N} - \hat{N}_j)}, \tag{2}$$

where $\hat{N}_i$ is the total number of ground-truth pixels for the i-th floor plan element in the floor plan. $\hat{N} = \sum_{i=1}^{C} \hat{N}_i$ represents the total number of ground-truth pixels over all the C floor plan elements in the task.

• *Cross-and-within-task weighted loss.* The within-task weighted losses for the room-boundary and room-type prediction tasks computed from Eq.(1) are denoted by L_{rb} and L_{rt}, respectively. Total number of network output pixels for room-boundary and room-type are labelled as N_{rb} and N_{rt}. The overall cross-and-within-task weighted loss L is defined as:

$$L = w_{rb} L_{rb} + w_{rt} L_{rt}, \tag{3}$$

where w_{rb} and w_{rt} are the weights given by:

$$w_{rb} = \frac{N_{rb}}{N_{rb} + N_{rt}} \tag{4}$$

$$w_{rt} = \frac{N_{rt}}{N_{rb} + N_{rt}} \tag{5}$$

3.2 Implementation Details

3.2.1 Datasets. As the floor plans held by the Municipality of Amsterdam were not made available for the project due to privacy concerns, the New York R3D dataset annotated with pixel-wise labels by Zeng et al. [10] was used for the project. The dataset contains 214 rectangular floor plan images and 18 images of round-shaped layouts. Additionally, 20 floor plan images, scraped from the Funda [1] website, were manually annotated in Photoshop and added to the dataset. Most rooms shapes in the R3D dataset are irregular with non-uniform wall thickness. For the train-test split ratio, the R3D dataset was randomly split into 179 images for training and 53 images for testing.

3.2.2 Tuning Hyper-parameters. Learning rate, the parameter that determines how fast or slow the network moves towards optimal weights, is one of the most important hyper-parameters for tuning neural networks [12]. The model requires on average 12 hours to complete 1 000 epochs of training. Achieving a desirable learning rate is important as it will reduce training time and decrease cloud graphics processing unit (GPU) costs. The goal was to tune the learning rate such that it is low enough that the network converges to a model that can make accurate predictions but high enough so that it can be trained in a reasonable amount of time. A systematic trial and error approach was taken in order to efficiently find an optimal learning rate. The starting point was a fixed learning rate of 1e-4, as leveraged by Zeng et al. [10]. Taking this number as a baseline starting point, a search was conducted from a coarse to a fine learning rate. The search consisted of testing learning rates at various orders of 10 to find an optimal range, moving in both directions from 1e-4. Through extensive trial and error testing, a satisfactory fixed learning rate was identified at 5e-4, as can be seen in Figure 2.

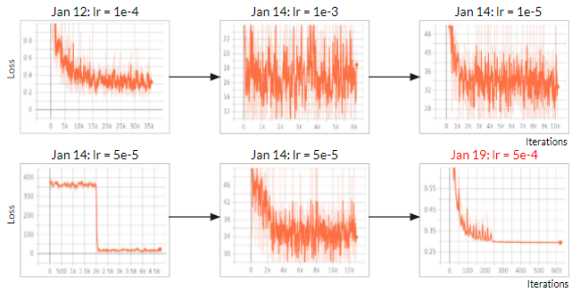

Figure 2: Searching for Desirable Learning Rate from Baseline Rate of 1e-4

3.2.3 Network training. The network was trained on an NVIDIA P5000 16 GB GDDR5X machine rented through Paperspace [2], as a GPU was required to train the CNN. The Adam **reference** optimizer, an algorithm for first-order gradient-based optimization of stochastic objective functions, was employed to update the parameters and a fixed learning rate of 5e-4 was used to train the network. A batch size of 1 was used as a batch size > 1 reduced the model's performance in making accurate predictions. A convergence counter was implemented to control epochs. If loss did not improve in 50 epochs, learning rate was dropped by $log 0.5$ and if loss did not improve in a further 75 epochs the loop was broken and the model was considered as converged. The step size, which controls how many epochs are allowed without a new best loss before decreasing learning rate, was set to 25 and gamma, which

[1] https://www.funda.nl/
[2] https://www.paperspace.com/

represents how much the learning rate is decreased at every step size was set at 0.3.

3.3 Evaluation

For quantitative evaluation, two commonly adopted metrics were used [13], namely, the overall pixel accuracy and the per-class pixel accuracy, defined as

$$overallAccuracy = \frac{\sum_i N_i}{\sum_i \hat{N}_i} \tag{6}$$

$$classAccuracy(i) = \frac{N_i}{\hat{N}_i} \tag{7}$$

where $\hat{N}_i$ and N_i are the total number of the ground-truth pixels and the correctly predicted pixels for the i-th floor plan element respectively. The focus of model evaluation is to determine how often the model is correct in correctly predicting pixels out of all the ground-truth pixels.

Comparing tuned model against AMS annotated model. To evaluate how the addition of new annotations impacts the performance of the base model in floor plan recognition, two separate training sets are evaluated. The first is comprised of the original New York R3D dataset, *Model_20_lr5-e4*, while the second contains the additional Amsterdam (AMS) annotated floor plan images, *Model_AMS3_lr5-e4*. Table 1 illustrates that *Model_20* has an overall accuracy of 77% while the *AMS3* model struggles to identify room types and as such, accuracy drops to 63%.

Table 1: Comparison of base model vs. AMS annotated model

	Model_20_lr5-e4	Model_AMS3_lr5-e4
Overall accuracy	0.7759	0.6328
Room type *(Mean accuracy)*	0.4267	0.1425
Room type + boundary *(Mean accuracy)*	0.5324	0.3075

3.4 Post-processing

3.4.1 Connected components labeling. When a test floor plan image is fed into the network, an output is generated. However, due to the per-pixel prediction, the output may contain noise. To mitigate this, the images are further processed to remove inconsistent pixels and increase model accuracy. The post-processing involves locating connected components based on room boundaries.

Connected components labeling is defined as the *"creation of a labeled image in which the positions associated with the same connected component of the binary input image have a unique label"* [14]. In the image graphs, connectivity can be a 4-connected neighbourhood or 8-connected neighbourhood [15], as illustrated in Figure 3.

After connecting neighbouring pixels to respective components, the number of pixels of different room types within the bounded area is counted. Then, the inconsistent pixels are replaced by the type with highest frequency within the region. Figure 4 illustrates that the minority predictions in green and yellow are reset as the majority prediction in orange after post-processing.

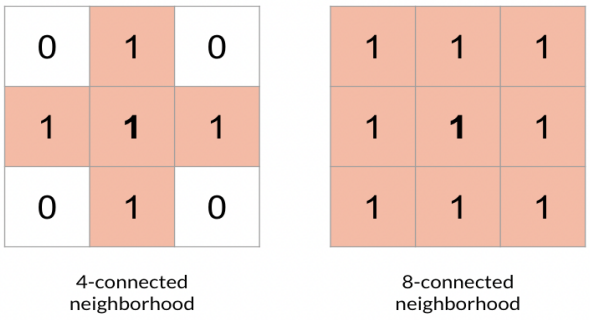

Figure 3: Connectivity in 4-Connected Neighborhood and 8-Connected Neighborhood

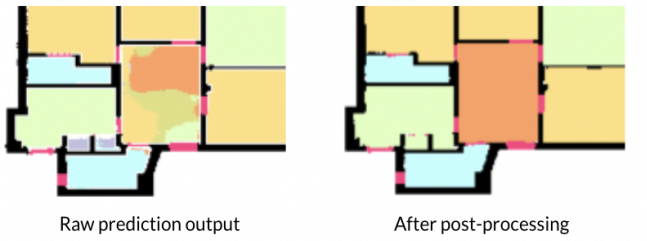

Figure 4: Floor Plan Prediction after Post-processing

3.4.2 Compartment counting. After labeling connected components, the number of different compartments, doors and windows is counted as long as the connected areas are larger than a certain size. The size is determined by a selected number of pixels. This measure can prevent some small and trivial regions, due to ambiguous predictions, from being counted as a single room or object. The number of pixels for different compartments, doors and windows is experimental and can be further optimised according to user context. For instance, when a connected region is classified as a door, it will only be counted as such if the area is larger than 80 pixels.

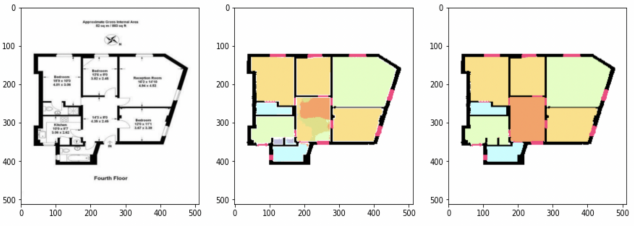

Figure 5: Original Floor Plan, Raw Prediction Output, and Prediction after Post-processing

4 INTERACTION DESIGN

4.1 Requirements

In order to understand the users of the system, their activities and the context of these activities, a set of key requirements was established. These are based on six types of requirements that help to shape the system in a way that it supports the users in achieving their goals [16]. These requirements provide a basis for making purposeful design decisions and identifying relevant usability testers. The six key requirements are functionality, data, environment, user characteristics, usability goals and user experience.

4.1.1 Functionality. The main purpose of the system is its ability to convert floor plan images to structured data. It should therefore be possible to easily feed floor plan images to the system and to extract the structured data from the system once the conversion is complete. Visual representations of the structured data should be generated for ease of interpretation, to see how the model evaluates individual floor plans.

4.1.2 Data. The data the system handles consists of floor plan images. The images should have a .jpg or .png datatype. These images can vary in resolution and in floor plan notation style. The system should be able to handle a wide range of these variables. The system converts the floor plan images to structured data. It is required that the output have a text or .csv format.

4.1.3 Environment. The physical environment the system is used in will mainly be on computers in the office buildings of the municipality of Amsterdam. It is also possible that some employees will use the system from a home computer, as they would during the COVID-19 pandemic. The social environment is a single data science team of the municipality of Amsterdam. It can be helpful to share the output generated by the system, therefore it is important that data is output in a common format. The data can be shared through other existing systems. Collaboration on the system itself is not necessary, as the team consists of few people.

The organizational environment of the Amsterdam DataLab ensures that communication infrastructure is well established and user support will good. The system will have to run in a technical environment that allows for processing images at reasonable speed. The technical requirements are elaborated upon in section 5.2.

4.1.4 User characteristics. The end users of the system are a team of the municipality of Amsterdam. The members of this team are data scientists. This means they are experts and that it can be assumed the end users have extensive experience working with computer programs and possess the ability to navigate them. Therefore elaborate instructions for the system are not necessarily required. However, many people with tech or programming experience greatly value a simple, intuitive and well thought-out design.

4.1.5 Usability goals. The system should perform a very singular goal, namely converting floor plan images to structured data. The utility of the system comes from the fact that the structured data can be used in other models to make predictions. Next to useful, the system should be effective. As there are not many functions, the system should not be overly complicated and perform only the actions that are required. Lastly, the stakeholders indicated that the floor plans they will use are private information that should not be publicly available. Therefore, it is important that the information entered into the system is safe and cannot be extracted in any other way. After processing the images the system should therefore remove the images from the server, or the system should run on a local server so it can only be accessed by the Municipality of Amsterdam.

4.1.6 User experience. The system is designed to aid in performance of another model by generating structured data. The most important requirement in terms of user experience is therefore that the system is experienced as helpful. Additionally, as stated under user characteristics, it is assumed that the end users value an intuitive and simple design. Accordingly, it is important that the end users feel that the system is intuitive and clear. In creating the design for the system it should be avoided to make it over-complicated or cluttered.

4.2 Competitive usability evaluation

A competitive usability evaluation is performed on two websites to identify what does and does not work well on these sites. This analysis is performed to guide the design of the first iteration of the prototype [17]. The form of this competitive usability evaluation is a competitive review, meaning that an in-depth look is taken at the two websites and relative strengths and weaknesses are identified.

The goal of a competitive evaluation is to look at what the competition is doing to improve the design of a system. In the case of the current system no direct competitors exist as there are no tools with the exact functionality of the system under development. However, there are sites that have a comparable function [3] and sites that have a comparable intended structure [4].

4.2.1 Comparable function. The Vision AI page can be found in appendix A. Vision AI is a Google Cloud service which offers two products: the first is AutoML Vision. This product provides automated training for custom machine learning models focused on image detection. The second product is the Vision API. The Google Vision API allows for the uploading of images. These images are

[3] https://cloud.google.com/vision/
[4] https://smallpdf.com/word-to-pdf

Figure 6: Uploading a floor plan image to the Google Vision API does not produce a relevant output

then labelled and classified by a pre-trained machine learning model through REST and RPC APIs. The model also detects faces, objects, printed and handwritten text.

Below the textual explanation of both products the site offers the option to use the API. It shows a box containing the text '*drag image file here or browse from your computer*'. When a file is uploaded a click captcha is shown. After completing the captcha, the results of the analysis are shown in four or five tabs, depending on the image that is uploaded. Clicking on a tab shows the tab information on the right and the uploaded image on the left. The first tab, 'objects', shows the objects that the model has identified and the probability with which these objects have been identified correctly in percentages. The objects are indicated on the image in a green rectangle. When hovering over a certain object, the rectangle is highlighted in the image. The second tab,'labels', shows the labels that have been assigned to the image by the model and the probability with which the labels have been assigned correctly in percentages. The third tab, 'text', shows the text that is present in the image. If there are multiple places where text is found, the text is organised by page, then block, and then paragraph. The fourth tab, 'properties', shows the dominant colours in the image in percentages and crop hints. The fifth and final tab, 'safe search', shows the likelihood of the image being adult content, a spoof, a medical image, a violent image, or a racy image on a 5 point Likert scale ranging from very unlikely to very likely.

Below the result tabs a drop-down button is present labelled 'show JSON'. When the button is clicked the request URL, request JSON code, and response JSON code are shown in separate text boxes. Each can be copied to the clipboard with the click of a button. Next to the 'show JSON' button, there are also two buttons labeled 'Reset' and 'New file'. The reset button reverts the page back to the state it was in before the image was uploaded, allowing for a new upload. The 'new file' button directly allows a user to select a new file from their computer to upload.

Below the introduction of both products and the try-out box for the Vision API, six pictograms accompanied by text are visible. These explain the functions and possibilities of the products in more detail. The remainder of the page below the pictograms is about customers, current use cases, pricing and other customer related information. This part of the site is not relevant for the current evaluation and will therefore be omitted.

In the Vision AI site, it can be seen that it is possible to have a site onto which images can be uploaded and which can then be analysed by a pre-trained model, through an API. This is essentially what the intended system should do. Another takeaway from this site is the ease with which a visitor is able to upload an image. It only takes two clicks or a single drag. It also convenient that a new image can be uploaded with the click of a button.

The results given by the models show that the model works well for general purposes. However, the website is not able to generate relevant output for more specialized tasks like identifying different features in floor plan images. This can be seen in figure 6, where a floor plan image was fed to the Vision API. When creating a specialized tool it is possible to display more relevant information. The positive side to the output is that it is made easy for the visitor to download the extracted information and to save the request.

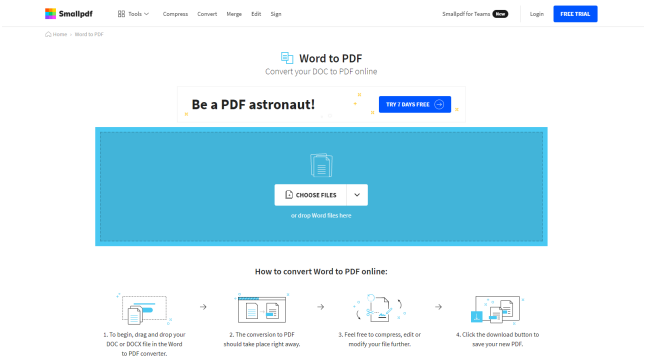

Figure 7: The page layout of Smallpdf is very functional and provides a good template for a conversion tool

Additionally, representing the findings of the model in the image makes it easier to comprehend for users.

4.2.2 Comparable intended structure. Screenshots of the Smallpdf site can be found in appendix B. Smallpdf offers a collection of PDF tools to process, compress or convert digital documents. The page layout and functioning for most of these types of tool pages is very similar. For the purposes of this evaluation, one of these tools is analysed. The initial screen of this tool can also be seen in figere 7. The Smallpdf Word to PDF converter does not have the same function as the intended system. However, the page is similar to the intended system in the sense that both offer a single function, which is to take a certain input, convert it and deliver a certain output. The Smallpdf site has a minimalist look and feel. The title explains what the function of the page is: '*Word to PDF - Convert your DOC to PDF online*'. It is followed by a big block labeled 'Choose Files'. Clicking it allows users to upload their intended Word file. It is also possible to drag Word files into the block.

The upload block is followed by a block explaining how to use the page. The four steps to extract results are explained with pictograms and text. Below this the key values of the organisation are summarised. Finally, links to several articles providing more information are included on the page.

When a file is uploaded, the site indicates that the file is being converted. After the conversion is complete a new page appears. On the left, the result of the conversion is represented. On the right, there is a large button that is labelled 'Download'. Through this button it is possible to download the file to several different locations. Above this button it is possible to adjust the name of the file that a user can download. Next to the download button, there is also a button to share the file through a link. In addition there is a button with an arrow labelled 'Start over'. Clicking it returns the user to the upload page.

The Smallpdf site is very functionally optimised. It is clear what the purpose of each page is. Sufficient information is provided but it is not overwhelming or cluttered. The upload button is the central focus point of the page, which makes sense because it is the main functionality. This also makes uploading easy. Another positive point is the processing animation. This makes the status of the system visible and the system provides feedback about this

status to the user, indicating that the action they performed has an effect[18].

The page presenting the results of the conversion is also very functional and clear. It provides the option to download, adjust the file name, and upload another file in a comprehensible fashion. For the purpose of this site, no further options are necessary and no further options are provided. In general, the interaction design of the Smallpdf site provides a good template for designing the floor plan tool.

4.3 Application Prototype Explorative Usability Test

4.3.1 Method. An initial click dummy was constructed based on the system requirements and the previously performed competitive analysis. This click dummy prototype was created using the tool Figma[5]. Screenshots of the Figma prototype can be found in appendix .... . The rationale for creating this first prototype was to determine whether the intended system design worked intuitively and clearly for users before the actual system was designed.

To test this, a remote usability test was conducted with 5 participants. Using 5 participants is usually enough to discover 80% of the usability problems[19]. The participants were selected from the MSc Information Studies at the University of Amsterdam (UvA), because it can be assumed that people following this programme would have similar technical skills and insights as the intended end users, who work as data scientists for the Municipality of Amsterdam. The actual end users were not used for this test, as there are only two.

Recruitment. The participants were recruited through WhatsApp groups for the MSc Information Studies and via Canvas discussion boards. Participants were invited to a Zoom video-call, which was recorded with permission from the participants. To ascertain whether participants were representative of the end users, they were asked a number of background questions. First, they were asked to indicate their age, to ensure participants were over 18 years old. Second, participants were asked to describe their academic background. This was done to confirm they are currently following the MSc Information Studies. Third, participants were asked to give an estimate of the number of hours they spend online per week, to further establish that their digital literacy was comparable to that of the end users. Finally, participants were asked to confirm whether they would be using a laptop for the entirety of the usability test.

Usability Test. After answering these questions, they received a link to a usability test, which was created in Maze[6], a tool specifically designed for early stage, remote usability testing. The test consisted of a welcome screen, an informed consent form, a context screen, a 5 second test, 3 missions to complete on the prototype and 3 open ended questions. During the test an observer was also present to ask for elaboration on the answers given by the participants. Participants were asked to share their screen during the test so the observer could examine their interactions with the prototype. The recorded interviews were transcribed and summarised post interview.

[5]https://www.figma.com/
[6]https://maze.co/

The Maze test started with a welcome screen that explained the reason for conducting the usability test and how the test functioned. Continuing to the next step brought participants to an informed consent form. Here participants gave permission to use the data gathered during the usability test. If participants selected 'no' to this question, the results of their test were removed and deleted. The informed consent form was followed by a context screen. Here, a short text explained what the prototype was meant for, who would be using it and what they would be using it for. If participants had questions, the observer could elaborate on this information. Participants were given this information to make their behaviour more representative of the end users, who would possess this prior contextual knowledge.

After the context screen, a 5 second test was performed. Participants were asked to try to understand what the image they were about to see image is about and remember as much information as they could. When they proceeded, they were shown the landing page of the prototype for 5 seconds. Afterwards, they were asked what their first impressions of the previous image were.

Missions. The 5 second test was followed by three missions. The participants were asked to think aloud during the missions, meaning that they spoke about the reasoning behind their actions during the missions. In the first mission, participants were asked to extract features from *floorplan 8_32.jpg* by uploading it. This was one of the floor plans present in the prototype. For the second mission, participants were asked to download the extracted features. The final mission was a multiple choice question. Participants were asked to indicate how many windows were present in the floor plan. Participants were asked to pause after each mission for potential follow-up questions by the observer.

After completing the missions, participants answered three open ended questions. In order, these were: "Do you understand what features are extracted from the floor plan?", "What do you think of the overall design of the floor plan feature extraction tool?", and "Can you think of some improvements to the current design?". The observer asked for elaboration and added follow-up questions where necessary. When participants finished answering the open ended questions a final screen was presented, indicating that the test had concluded and thanking participants for their collaboration. The observer then answered any questions the participants had, after which the recording was stopped and the call was ended.

4.3.2 Results. The recordings of the usability tests were transcribed in a clean verbatim format. In this manner, participant wording is preserved as precisely as possible, but the text is lightly edited for readability. Additionally, personal and irrelevant information, such as the explanation of thinking aloud and sharing the link via Zoom, is omitted. The transcripts of the usability test can be found in appendix. In this section the most important insights gained from the usability test are summarised.

5 second test. The 5 second test showed similar results among all participants. All participants noted that there was a large emphasis on the upload button. This makes sense as it features prominently on the landing page and uploading floor plan images is the main function of the tool. It was also immediately clear to 4 out of 5 participants what the purpose of the page was, indicating that the design of the landing page serves its function.

First mission. The first mission, uploading a floor plan image, was completed successfully by all participants. Every participant used the optimal path. Participants generally noted that the actions they had to take felt intuitive and that the interface was quite clear. One point of feedback from the first mission is that some participants indicated that the page looked like you could drag your files onto the upload button to upload them. Such a feature may be a useful addition on the final site.

Second mission. The second mission, downloading the extracted features, was also completed without any problems by all participants. Every participant used the optimal path. To 4 out of 5 participants the button was clear. However, one participant indicated that he felt the label 'download data .csv' was a bit odd. According to him it would make more sense if the label was more descriptive, as the label 'data' was a bit vague. He suggested calling it 'download extracted features as csv file'. Besides that, the function seemed completely clear to all participants.

Third mission. The third mission, indicating the number of windows present in the floor plan, was completed successfully by 4 out of 5 participants. Here it could be observed that most of the participants were inclined to start counting the windows in the floor plan image. This resulted in one participant filling in a wrong answer. However, the participants that started counting also indicated that they did so because of the form of the test. The formulation of the question and the form of the interface made them think they had to do this manually. Upon looking a bit longer at the page, 4 out of 5 participants did notice the text box containing the extracted features. Most of them indicated that it did not really catch their attention and that it was a bit unclear.

Open ended questions. The open ended questions led to both positive and negative feedback. Participants generally indicated that they thought the design was very functional and clean. Their attitude to this was quite positive. Everything were where they thought they would find them and functioned as they anticipated. Participants were also positive about the 'what it is' page. This was possibly owing to their academic background or because it explained some things they could not find on the output page.

The major point of criticism across all participants was the output page. To many it was not completely clear and the data presentation should be a focus point for the next iteration of the system. All participants had some issues comprehending the images. According to 4 out of 5 participants these issues could easily be solved by adding a legend containing the colours used in the output image. In a way this legend was already present in the text block. However, some participants indicated that there was too much information in the extracted features text block or that it was not presented clearly. Improving the text block structure and making the text larger were offered as possible solutions.

Lastly, 2 out of 5 participants indicated that they thought the design did not feel very reliable. They said they might worry about the security of the data they upload. According to them it might help to put more information on the landing page. The reliability might not be a completely relevant concern, as the tool will be used by the stakeholder in a secure environment. Increasing the amount of context given before taking the usability test can decrease these concerns.

In summary, the explorative usability test on the click prototype indicated that the design is clear and intuitive to use. For the second iteration of the prototype no large adjustments have to be made to the structure of the site. However, the most pressing problem is the output page. This page contains important information, but is not clear to most users. The page design of the page will have to be adjusted in the second iteration of the prototype.

5 SYSTEM DESIGN

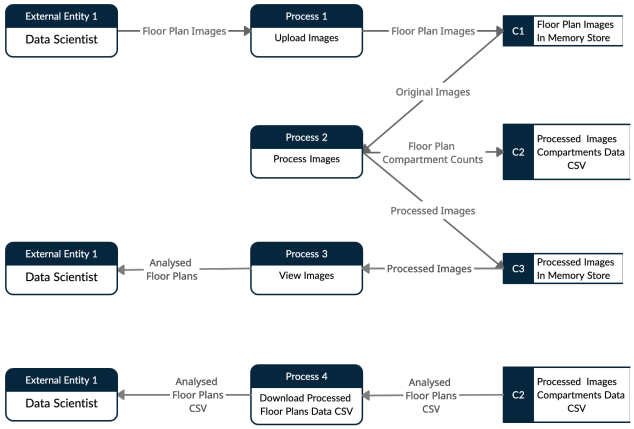

Figure 8: Data Flow Diagram

5.1 Development Process

The design of the system was driven by the necessary transformations and processing of the data, as depicted in Figure 8. Three distinct user interactions were identified, along with the automatically triggered generation of the structured floor plan data. The necessary data structure for each interaction was determined, and a corresponding application programming interface (API) was defined. The strict definition of each API structure allowed for the parallel development of the frontend using the API, and backend implementing it.

APIs were defined for the upload of the floor plan image, the retrieval of the analysed floor plan image, and the retrieval of the analysed floor plan CSV file. The analysis of the floor plan is triggered automatically when a user uploads the floor plan image, so no API was necessary for this task.

5.2 Application Architecture

The system architecture is illustrated in Figure 9. An API handler receives requests from the web UI, calls the corresponding function(s), and returns a response to the UI. Requests to upload images use a HTTP POST request, which will store the images in memory, before triggering the image processor. Images are kept in memory to make accessing them faster and to execute fewer CPU instructions. The image processor will generate images of the analysed floor plans with compartments identified and a CSV file with the extracted features. Requests for images and CSV data use a HTTP GET request. The request is resolved by returning the processed floor plan data from in-memory.

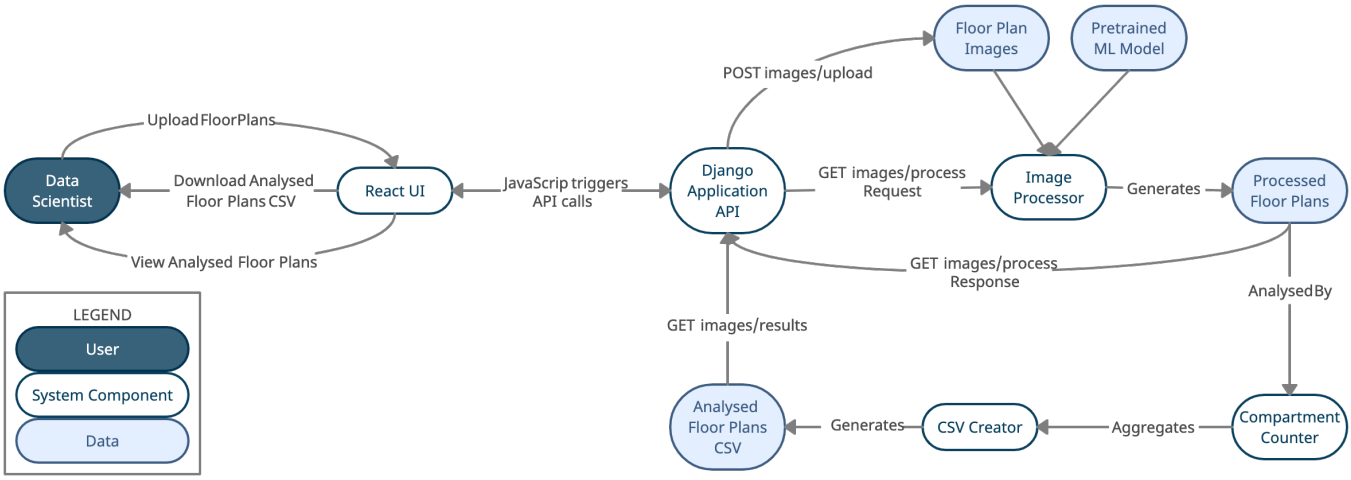

Figure 9: High Level System Architecture

The frontend was initially prototyped using Figma, this prototyping phase resulted in a frontend design which was re-implemented in JavaScript. Screenshots of the final UI can be found in appendix D. The JavaScript frontend was built using the React framework [20]. React is a component driven framework which updates document components individually, when source data is modified. This means that dynamic pages can load quickly and efficiently. The React code was optimised for performance using a process called minification [21]. Minification is the transformation of human readable JavaScript source code into a smaller file, so that it can be downloaded faster.

The React frontend only renders documents for web browsers, and interacts with a backend web service which handles the logic of the system. Django [22] is a highly popular Python web framework for writing REST web services. REST [23] is an industry standard architecture for interacting with web services. Django automates the generation of boilerplate code, so that developers can focus on the logic of the overall system. Python is a dynamically typed high level language designed for rapid prototyping. This means developers can develop working systems quickly without needing to implement common data structures and functions. For these reasons, the backend web service was built using the Django Python framework.

6 FINAL USER INTERFACE USABILITY TEST

6.1 Method

The initial click dummy and the results of the first usability test were used to create a functional prototype. The system design of this prototype is elaborated upon in section REF SECTION. Screenshots of the functional prototype can be found in appendix A. To test if users were able to perform the intended tasks with this prototype, another remote usability test was conducted with 5 different participants from the first group, so that testing results would not be skewed by prior knowledge gained in the first test. Participants were selected from the MSc Information Studies and the MSc Artificial Intelligence at the University of Amsterdam, to ensure participant technical proficiency aligns with the intended end user.

6.1.1 Recruitment. Participants were recruited directly via WhatsApp messages. Participants were invited to a Zoom video-call, which was recorded with their permission. The tasks participants had to perform were shared in the Zoom chat, so participants could check the chat if they forgot the task. To determine that participants were representative of the end users, the same background questions as were asked in the Explorative Usability Test were posed.

6.1.2 Context. After answering these questions, the observer provided participants with the context of the project. The observer explained that the municipality is working on a model that is able to predict fire risk in commercial buildings and that this tool was created to extract structured data from floor plan images of the buildings, because that structured data could then be used in the fire risk model. The observer also asked participants to pretend that they were employees of the municipality who were working on this model. Participants were intentionally given more information than in the first usability test to ensure that participants understood their actions and so their behaviour would be more representative of the end users.

6.1.3 Usability Test. Participants then received a .zip file containing 16 images from the Funda web scrape. They were asked to download these images to their computer, because they would need them during the test. After downloading the images, participants received a web-link to the prototype. Participants were asked to open the link and share their screen. When participants started screen sharing, the observer shared the first mission in the chat. Participants were asked to think aloud during the missions. Participants received 2 more missions and another 3 open ended questions. The observer indicated to participants when they had successfully completed a mission and asked for elaboration on answers provided, where necessary. The recorded interviews were transcribed and summarised.

6.1.4 Missions. For the first mission, participants were asked to upload one of the floor plan images provided to them. Participants completed this mission by uploading one of the floor plan images, which brought them to the output page. For the second mission, participants were asked to download the features that were extracted from the floor plan. Participants completed this mission by downloading a .csv file with the extracted features. For the third mission, participants were asked to upload a batch of at least 3 floor plan images. To complete this mission, participants had to return to the homepage and then upload at least 3 floor plan images at once.

After completing the missions, participants answered three open ended questions. In order, these were: "Do you understand what features are extracted from the floor plan? Please explain what you see as well as you can.", "Please navigate the site freely. Click around for a while, take a look at the pages you haven't seen yet and explain what stands out to you.", and "Do you feel like the site is reliable?". The observer asked for elaboration and added follow-up questions where necessary. When participants finished answering the open ended questions the observer then answered any questions the participants might have and thanked them for their participation. The recording was then stopped and the call was ended.

6.2 Results

Similar to the usability test using the Figma prototype, the recordings of the second round of usability tests were transcribed in a clean verbatim format. The same light editing was performed on the transcripts as detailed in the first usability test. The transcripts of the usability test can be found in appendix .... . In this section the most important insights gained from the usability test are summarised.

6.2.1 First Mission. The first mission, uploading a floor plan image, was completed successfully by all participants. Every participant used the optimal path and misclicks were minimal. Again, participants generally noted that the actions they took felt intuitive and the interface was clear. Similar to the explorative usability test, 2 out of 5 participants indicated that the interface looked like it would support dragging files over the button and dropping them to upload the images. As this is a recurring point, it would be good to make dragging and dropping an option for uploading, or to adjust the design so it no longer makes users feel like this is an option. Additionally, one participant noted that he would like to see a confirmation button before the uploaded images would be processed. He explained asking confirmation before analyzing would be especially useful when uploading a larger amount of images.

6.2.2 Second Mission. The second mission, downloading the extracted features, was also completed without any problems by all participants. Every participant used the optimal path. However, when more than one image was uploaded it was not clear to 2 out of 5 participants whether the download button generated a .csv file for the floor plan they were currently viewing or for all the floor plans that they uploaded. Changing the label of the button to 'Download extracted features for all floor plans' would likely solve this problem. Another point that was noted by one of the participants was that it might be helpful to include user feedback for the downloading of the extracted features. Adding an icon similar to the 'analyzing' icon that is shown when uploading an image, would suffice. Finally, 2 out of 5 participants noted that it might be useful to make the output image downloadable.

6.2.3 Third Mission. The third mission, uploading a batch of at least 3 floor plan images, was also completed without any problems by all participants. However, it should be noted that 3 out of 5 participants indicated that they might not have thought that it would have been possible to upload multiple images at once had they not explicitly been told to do so. Most participants noted that they likely would have figured this out, but it would be helpful if it were clearer. This result is important as it shows that one of the main functionalities of the system is not immediately clear.

6.2.4 Feedback. The majority of criticism given during the open ended questions was again directed at the output page. This shows that the improvements made from the Figma prototype were not sufficient. The 'features extracted table' still attracts little to no attention and at first it is not clear to most users what the colours in the output image represent. According to 3 out of 5 participants, adding a colour legend for the output image would help, especially if this legend was right next to the image instead of below it, as users would not have to scroll up and down to see it. Additionally, 3 out of 5 participants also indicated that the images were a bit small. They suggested either implementing a zoom function or increasing image size.

An unexpected amount of feedback was given about the scroll bar. This is possibly due to the fact that it was something that was in no way included in the Figma prototype. According to one participant it would be helpful to always show the scroll bar, instead of only showing it upon interaction. This makes it less likely that users will not see that there is additional output. Another user indicated that it would be useful to highlight the image that is currently selected. This would make navigation easier. Additionally, he said it might help if the scroll bar included a file name. Yet another user voiced some concerns about the scalability of the scroll bar, as scrolling over a large number of images might prove cumbersome.

In contrast to the usability test using the Figma prototype, no reliability concerns were raised even though there was a question specifically concerning reliability. This is likely due to the larger amount of context participants received. Finally, 2 out of 5 participants indicated that they would like to see more feedback when the system is analyzing floor plan images. This could be achieved by having a moving image or showing how many of the uploaded images have been processed (e.g. 3/11 images analyzed). This would help to show that the system is actually working and not hanging.

The most important results of the second round of usability testing is the fact that it is not immediately clear to users that it is possible to upload several images at once and the system feedback that seems a bit lacking. These results matter most as they directly influence the functionality of the system. Therefore, the focus should lie on solving these issues. Improving the comprehensibility of the output page by adding a legend should be the second focus. A third, minor point of attention is the improvement of the scroll bar, as this does not greatly influence how the system is used.

7 DISCUSSION

The system successfully transforms unstructured floor plans to structured data, which can be a supplementary input to the fire risk model. The best-performing model achieves an accuracy rate of 77% in room boundary detection, and counting compartments.

In supervised machine learning, the model performance in terms of generalisation is highly dependent on its training data. A rich training set with diverse features can empower a model to generalise well and recognise floor plans with different styling format. In this research, some experimentation was done with adding some of our own annotated floor plans, however the model accuracy dropped, especially in room types, when the training data is composed of floor plans from both Amsterdam and New York. One of the reasons could be attributed to the variances in formats, styles and languages. In the R3D dataset, the floor plans in New York are purely drawn in black and white, while in the Dutch ones, the rooms tend to have a colourful styling. As a result of imbalanced samples and inconsistent styles, the model did not differentiate room types well.

7.1 Limitations

As the current model is only trained on residential floor plans, it might encounter underfitting for commercial floor plans. Additionally, the model may struggle with heavily annotated images, such as those with bounding boxes or further guiding lines that are not part of the actual floor plan. The model is also not able to correctly process multiple floor plans inside the same image file, such as multiple floors of the same building. These need to be input into the model separately. As input, currently only the image file formats JPEG and PNG are supported. Pertaining to user testing, there are limitations inherent in conducting remote tests, especially in qualitative testing. However, given the present pandemic situation in the Netherlands this was the only available avenue. It is further recognised that user participants who are representative of the end user, are not the end user themselves and this is an additional limitation of the user testing conducted in this project.

7.2 Future Work

The current iteration of the model is not able to differentiate between doors and windows as there is no differentiation between them in the training data. In the future the model could be modified to accommodate this differentiation and deliver a more precise output with doors and windows counted separately. This is not likely to negatively impact accuracy as the annotation styles for both are quite distinct. However, the training data would also need to be modified to include this differentiation. As this is a machine learning model, it can be further improved by training it on more diverse data, or more data that is in accordance with the guidelines of the building regulations of Amsterdam. Since the input data must be fairly clean for the model to work, a prepossessing step may be added to standardise differently formatted floor plans, such as those with heavy annotations outside the floor plan drawing. By cropping the image to exclude these annotations the model can be made to work for these images. Originally, the model was meant to include an estimation of the area of the compartments. This however proved to be unfeasible due to the scarce and inconsistent nature of such information in the training data. It may however be possible to include OCR ("optical character recognition") to extract this information if it is present in the source data. The model may also be modified to process PDF files, as well as other image file types, to enable a wider, more convenient use.

8 CONCLUSION

This paper proposes a method for extracting structured data from unstructured floor plan images, so this data can be leveraged as input features for the existing fire-risk model designed by the Municipality of Amsterdam. There are three key contributions in this work. First, the neural network hyper-parameters are tuned with an improved learning rate and a convergence counter which reduces model training by 50% from the baseline of 12 hours to 6 hours for model convergence. Second, the trained model is able to predict room-boundaries and room-types with an overall accuracy of 77%. This is lower than the 90% overall accuracy achieved by Zeng et al. [10] and indicates that there is opportunity to improve model architecture in future work. Finally, a system was designed that allows for easy and user-friendly access to the model, so that the end user is able to extract structured data from batches of floor plan images. The design of the system was based on a set of interaction requirements and a competitive usability evaluation. The functioning of the system was judged on the basis of two rounds of usability testing.

REFERENCES

[1] Salvage levert cijfers miljoenenbranden, 11 2018.

[2] Fire safety measures and permits | Business.gov.nl.

[3] Sterke groei in steden en randgemeenten verwacht, 2019.

[4] Bijna 71 duizend nieuwbouwwoningen in 2019, 2020.

[5] George Hadjisophocleous. FiRECAM System Model Documentation. 1996.

[6] J. Kaiser. Experiences of the Gretener method. *Fire Safety Journal*, 2(3):213–222, 3 1980.

[7] Sheraz Ahmed, Marcus Liwicki, Markus Weber, and Andreas Dengel. Improved automatic analysis of architectural floor plans. In *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pages 864–869, 2011.

[8] Chen Liu, Jiajun Wu, Pushmeet Kohli, and Yasutaka Furukawa. Raster-to-Vector: Revisiting Floorplan Transformation. Technical report, 2017.

[9] Samuel Dodge, Jiu Xu, and Bjorn Stenger. Parsing floor plan images. In *Proceedings of the 15th IAPR International Conference on Machine Vision Applications, MVA 2017*, pages 358–361. Institute of Electrical and Electronics Engineers Inc., 7 2017.

[10] Zhiliang Zeng, Xianzhi Li, Ying Kin Yu, and Chi-Wing Fu. Deep Floor Plan Recognition Using a Multi-Task Network with Room-Boundary-Guided Attention. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-October:9095–9103, 8 2019.

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 2015.

[12] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology, 8 2018.

[13] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. Technical report, 2015.

[14] Azriel Rosenfeld and John L. Pfaltz. Sequential Operations in Digital Picture Processing. *Journal of the ACM (JACM)*, 13(4):471–494, 10 1966.

[15] R. Fisher, S. Perkins, A. Walker, and E. Wolfart. Image Analysis - Connected Components Labeling, 2003.

[16] Jennifer Preece, Helen Sharp, and Yvonne Rogers. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons, 4 edition, 2015.

[17] Amy Schade. Competitive Usability Evaluations: Definition, 12 2013.

[18] Jakob Nielsen. 10 Usability Heuristics for User Interface Design, 11 2020.

[19] Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, 1 edition, 1993.

[20] React – A JavaScript library for building user interfaces.

[21] Optimizing Performance – React.

[22] The Web framework for perfectionists with deadlines | Django.

[23] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, Irvine CA, 2000.

A APPENDIX - GOOGLE VISION API

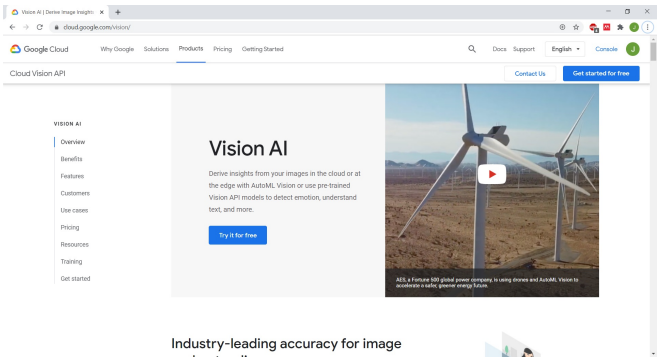

Figure 10: Competitive usability evalutation - Google Vision API screen 1

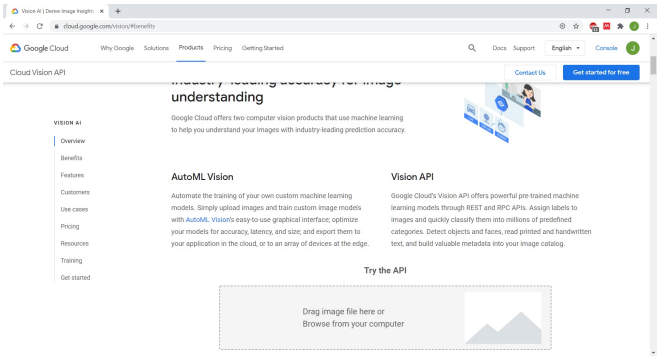

Figure 11: Competitive usability evalutation - Google Vision API screen 2

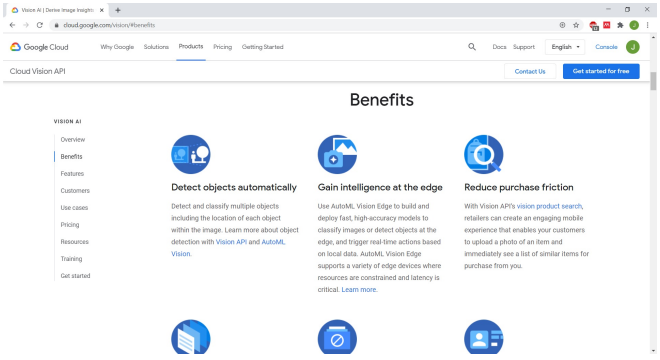

Figure 12: Competitive usability evalutation - Google Vision API screen 3

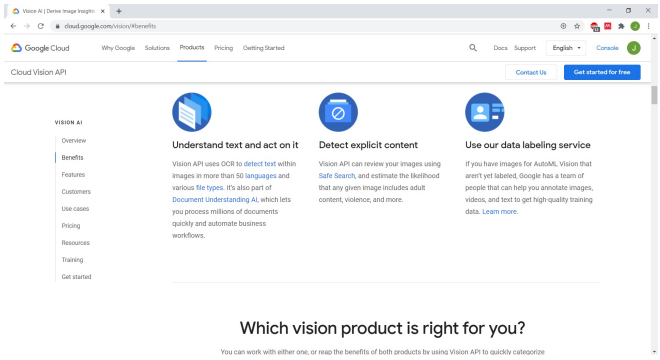

Figure 13: Competitive usability evalutation - Google Vision API screen 4

B APPENDIX - SMALLPDF

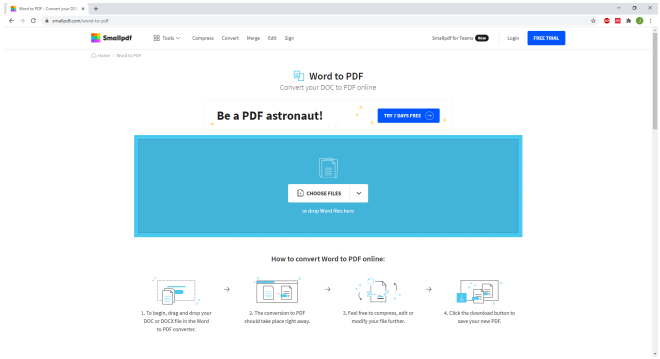

Figure 14: Competitive usability evalutation - Smallpdf word to pdf converter screen 1

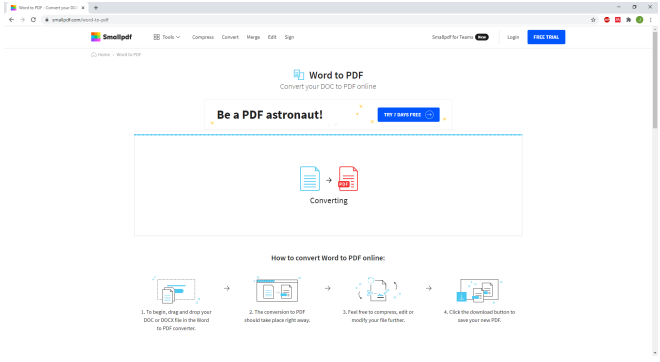

Figure 15: Competitive usability evalutation - Smallpdf word to pdf converter screen 2

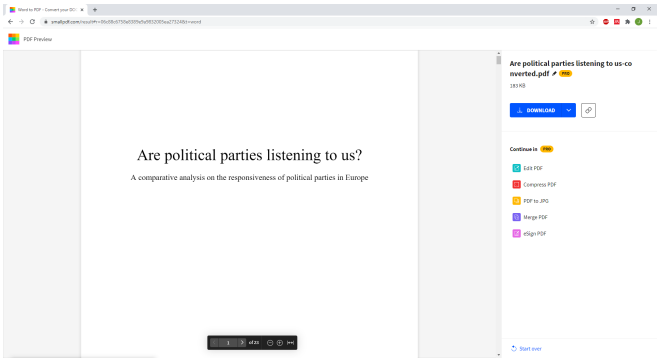

Figure 16: Competitive usability evalutation - Smallpdf word to pdf converter screen 3

C APPENDIX - APPLICATION PROTOTYPE

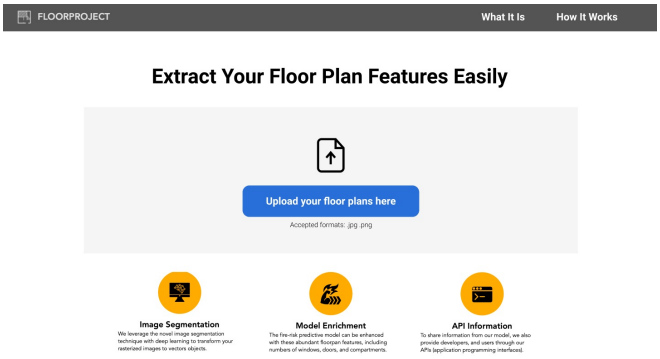

Figure 17: First prototype of landing page in Figma

Figma is a UI prototyping tool. The image shows initially designed landing page, where a user is prompted to upload floor plan images.

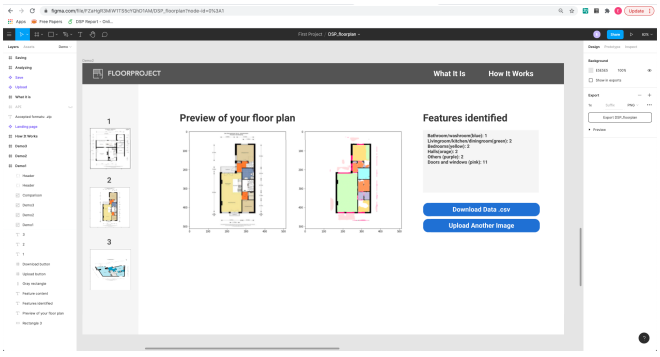

Figure 18: First prototype of results page in Figma

Figma is a UI prototyping tool. The image shows initially designed visualisation of analysed floor plans.

D APPENDIX - FINAL USER INTERFACE

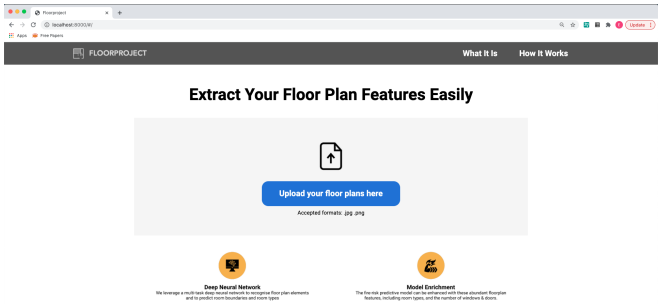

Figure 19: Landing page

This is the final implementation of landing page, implemented in React. On this page a user can upload floor plan images for processing.

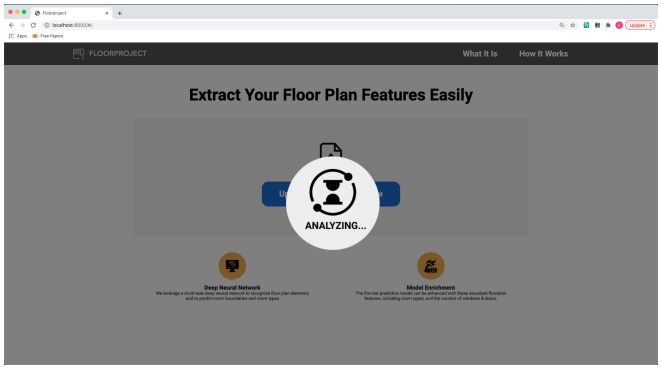

Figure 20: Loading screen

Loading icon indicates that floor plan images are being analysed and that uploading images was successful.

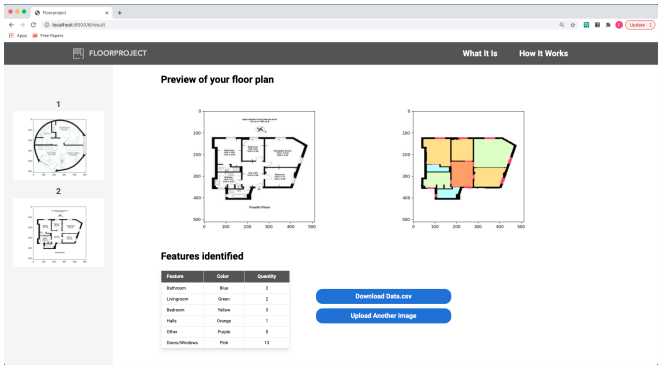

Figure 21: Extracted floor plan data

Results page visualises extracted info from floor plans for each image individually. Table lists compartment counts and it's possible to download all extracted data by clicking 'Downloading Data.csv'.