General Domain Neural Networks for Specific Image Similarity Domain

Using the Danish Real Estate dataset

Author:
Peter Friborg Jacobsen
wtk866

Supervisor:
Desmond Elliott

Datalogisk Institut
Københavns Universitet

January 18, 2021

Abstract

This study investigates the ability of two general-domain CNN's to perform as well as an in-domain model through image augmentation and fine-tuning the top layers of the general-domain models. The purpose of this study has been to teach two publicly available pre-trained general-domain models to be able to give meaningful representations of images from the Danish housing market, with only limited labeled in-domain images. This was done for the purpose of helping real estate agencies to increase their customer engagement, by allowing the customers to search for homes that matches their aesthetic preferences.

An elaborate investigation of transfer learning has been used to train different layers of the general-domain models. Through supervised transfer learning it was possible to teach the two general-domain models to be equally as good at representing features found in the in-domain images as the in-domain model were.

A previous study claims the impact of using the in-domain preprocessing scheme can improve a models top-1 accuracy by 30% points. In the purpose of recreating the results from the previous study, a variety of experiments were conducted, however this claim was refuted.

By reducing the dimensionality of the representation/feature vector it is possible to obtain higher top-1 accuracies, this was also reported in the previous study. The experiments in this study can acknowledge the impact of reducing dimensionality of the feature vector.

Finally an experiment with supervised contrastive learning is conducted, however due to the limited in-domain data available, this was not a success.

Contents

1 Problem Definition

Is it possible to close to gap between a general-domain pre-trained convolutional neural network and an in-domain pre-trained convolutional neural network, through the use of different fine-tuning ideas?

For the purpose of investigating the problem definition, a previous study by ***Ingwersen***[1] are used as threshold for the in-domain accuracies. The general-domain models proposed in this study are two convolutional neural networks pre-trained on the well known ImageNet dataset, and fine-tuned by training on a limited supervised in-domain dataset, originating from the Danish housing market. The domains of the general-domain and the in-domain are investigated to get an understanding of what learned features each model might possess.

2 Introduction

The purpose of this study is to increase customer engagement with advertisements of real estate in Denmark. This should be done by helping people find apartments which are similar to other apartments they like. The main problem is to identify rooms with a similar feature vector in the "Danish Real Estate" dataset.

A previous study, by ***Ingwersen***[1], found that in-domain pre-training (Places365) of a CNN produced big improvements in image similarity compared to a general-domain model pre-trained with (Imagenet). The central question of this study is to investigate the capability of the general-domain to close the gap to the the in-domain model with only a small amount of in-domain labeled data. Ideas include fine-tuning different layers of the CNN, and an implementation of the supervised version of contrastive learning.

2.1 The Problem Domain

The average person in Denmark is moving a total number of six times during their entire life[1]. The process of moving would by most people be equivalent to making a huge life decision. One obstacle home seekers will encounter is the overwhelming amount of houses on the market. Therefore filtering away uninteresting houses is crucial. This sound like an easy task for home seekers to overcome. However the filters available to the home seekers are very factual and only apply to hard constants like square meters, location and price.

A factor often considered essential when buying a home is the aesthetics of the home. Filtering for specific aesthetics is not available through any of the existing real estate agencies. An easy solution for real estate agencies would be to add a parameter: *Appeal* or *condition*, rating from a scale of 1 through 10. This approach face more than one obvious problem:

1. Who should rate the homes?
2. Everybody has different preferences

[1] https://www.boligsiden.dk/nyheder/2018/06/flyttestudie-vi-flytter-som-aldrig-foer/

These two problems can be conquered by having the home seekers mark homes which matches their unique home aesthetics. Then a model, like the one proposed in this study, would be able to identify homes on the market, that matches the home seekers unique preferences. The models proposed in in this study calculates the similarity between images, by measuring the distance between the encoded representations produced by the general-domain models.

2.1.1 Problem Solution

Throughout the progress of this study, an in-domain dataset, which will be examined in section 4, has been available to study. The general-domain models used in this study will through fine-tuning the top layers with the in-domain dataset be able to learn representative feature vectors. By comparing the feature vectors it will be possible to differentiate images that are similar to each other from images that are dissimilar. This idea is already used by companies in other industries, like `Pinterest` shown below.

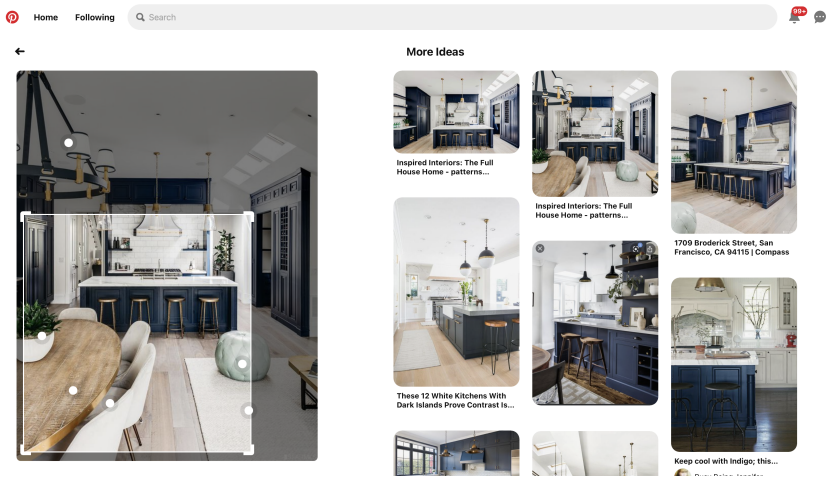


Figure 1: Pinterest Visual Similarity

This study has implemented different models, which like `Pinterest` are able to measure similarities in images. `Pinterest` is a big company and has a lot of resources available, which are not available to any of the real estate agencies in Denmark. The models in this study is an attempt to fairly cheap, using limited data and publicly available libraries and pre-trained models, to implement a solution for capturing similarities in rooms, which is similar to the model used by `Pinterest`. As the results will show, this was a success.

3 Background

Throughout this section the theory of the key ideas of the experimental setup is explained. This section will enlighten the basics of neural networks, the image specific convolutional neural network, transfer learning which were used to fine-tune the top layers of the CNNs proposed in this study, ideas in contrastive learning that could prove

to be a huge improvement in reaching the objective of the problem definition, and the similarity measurement which is used to describe how well the models perform.
Jump to sections Neural Networks, Convolutional NN, Transfer Learning, Contrastive Learning and Measuring of Similarity.

3.1 Neural Networks

Before progressing through the report the basics of neural networks should be understood. Neural networks comes from the idea of being able to teach a computer how to process an input, fundamentally similar to a brain, but without doing any programming (i.e. `If..else statements`). The human brain consists of 100 billions neurones, the neurones are what make the humans capable of interpreting signals from the outside world. The neurones are able to adapt themselves to react different if a certain scenario happens. The neural networks also uses neurones to interpreting inputs:

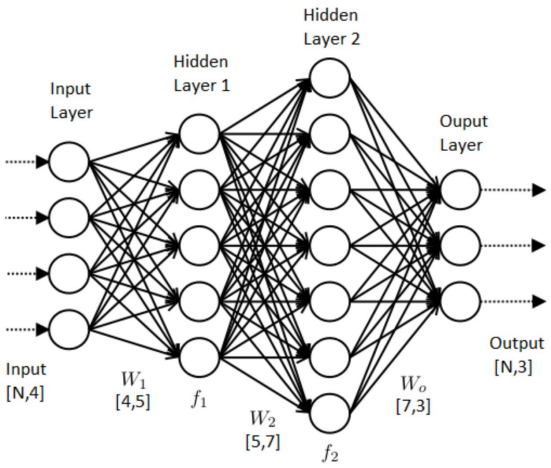


Figure 2: A simple neural network consisting of an input layer, two hidden layers, and an output layer

The figure, Figure 2, show a simple neural network, that consists of an input layer, two hidden layers and an output layer. Each layer consist of a various numbers of neurones, each neurone in a layer is connected to all the neurones in the next layer by weights, the weights define what information the neurone should forward to the next layer of neurones. Neural networks often involve adaptiveness of weights, which means that the neural network will be able to learn new ways of interpreting inputs. The weights are adapted through a learning algorithm, and a target value. The target value is what the neurone should accomplish, and the learning algorithm is how the weights should the updated, if the target is not achieved. All subsequent inputs will be processed by using the updated weights.

All models are wrong, but some are useful.

GEORGE E. P. BOX

3.2 Convolutional Networks

The convolutional neural network[2], denoted CNN, is a specific construction of neural networks, which has proven to be very useful when working with images. Convolutional networks uses convolutional filters between each layers. Convolutional networks have the advantage of being able to interpret the relationship between integers for all axes on each layer. This is what makes them suitable for understanding the relation between pixels in images. On the lowest levels of the convolutional network, the CNN will learn how to recognise edges and simple shapes. The higher the level of layers the more abstract and complicated the shapes. Below, Figure 3, it is shown how the convolutional filters work on each layer. Each layer of the network changes dimensions, so new convolutions can be applied and more complex shapes can be identified.

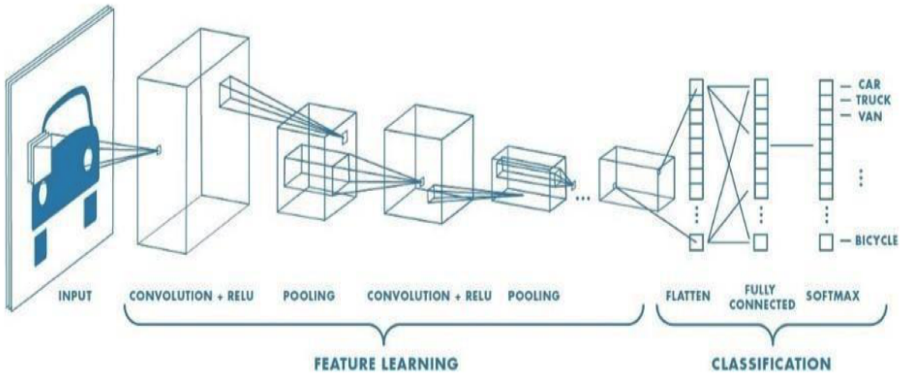


Figure 3: Architecture of a Convolutional Neural Network.
Image from: https://arxiv.org/pdf/2101.00793.pdf page. 8

Below are shown different convolutions applied to one of the images in the MNIST dataset[2], picturing the number 7. From these representations the CNN is able to learn different ways of interpreting this picture, the model is concatenating the interpretations from the convolution layers in the pooling layers.

[2] http://yann.lecun.com/exdb/mnist/

Figure 4: Different convolutions applied to the mnist dataset.
Image from: https://sempwn.github.io/blog/2017/04/06/conv_net_intro

The above case is simple compared to the object of this study, which is to measure similarities in different rooms, however the procedure is the same. Identifying objects in rooms and map the objects to specific labels, for example: a sink for kitchen, a bed for bedroom or couch for living room. These are the very obvious cases, some of the images of kitchens don't include sinks, and then hopefully the model has learned other representations which evaluates to kitchen. When training a network from the ground and up, a lot of data is required, and often data is very expensive to acquire. To accommodate the need for large and hence expensive data sets, other methods such as transfer learning can be applied that are less dependent on large data sets.

3.3 Transfer Learning

Transfer learning is a well-known and often used strategy when it comes to the task at hand of this report, i.e. classification. ***Karthik E***[3]s report suggests that high accuracy classification can be achieved when using transfer learning compared to both traditional machine learning used in a previous paper by ***Karthik E*** and a baseline model using only the source domain model with no training. The basics of the transfer learning task is to retrieve learned features from an existing model with a specific domain. The retrieved features can then be mapped together with the target domain, in the case of this project together with the Danish Real Estate Dataset. ***Pan Yang*** [4] introduces a definition for transfer learning, from this definition it is possible to categorise the transfer learning task faced in this project. In the paper two different cases are described, the first case where the feature spaces of the domains are different, and the second case where the feature spaces are the same, but the marginal probability is different. In this project, the first case is the proper one to use. Both images and labels have been available in the completion of this project, and therefore the inductive transfer learning implementation is chosen.

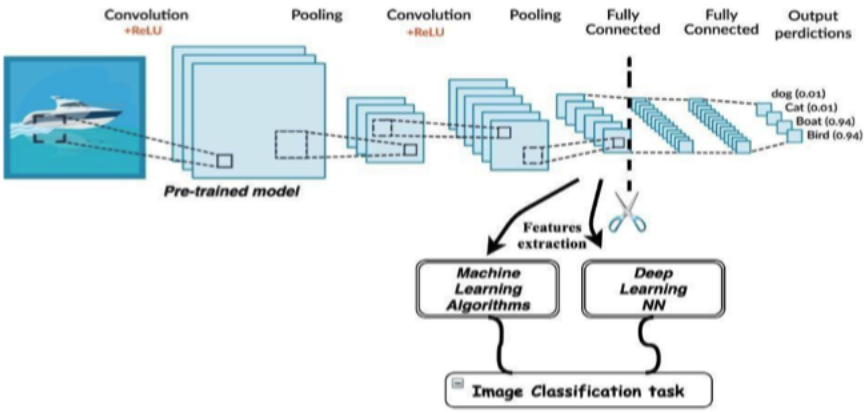


Figure 5: Transfer Learning applied to a CNN.
Image from: https://arxiv.org/pdf/2101.00793.pdf page. 10

The figure above, Figure 5, show how the transfer learning task is applied to a convolutional neural network. The pre-trained classification layer is discarded, the features are extracted and another classification layer is trained on top of the pre trained model. The pre-trained model can distinguish between four different categories, dog, cat, boat and bird. In the transfer learning stage, it will be possible to create multiple completely new categories, for example: motor boat, sail boat, row boat, canoe and a raft. Because the pre trained network know how to represent a feature space of a boat, it might be possible to teach it how to distinguish between different kinds of boats.

As shown later it is possible to implement this technique to train two general-domain models, Resnet50 and EffecientNetB0, to accurately categories and represent the feature spaces of the rooms of the Danish Real Estate dataset.

3.4 Contrastive Learning

For the purpose of finding similarities in rooms, ideas in contrastive learning have been investigated. Contrastive learning introduced two versions, unsupervised and supervised. Both of these versions have interest with regards to the creating a home recommender for home seekers. The unsupervised for the obvious reason that there is a continuous flow of new homes entering the real estate market, and it would be costly to label the new rooms through supervised measures. The supervised model is ideal for fine-tuning the model to improve the feature space similarity.

3.4.1 Unsupervised Contrastive Learning

Chen[5] introduces a state of the art unsupervised model implementation, which outperforms all previous unsupervised models by >5% on the ImageNet test set. Results from the article suggests the key to achieve the best model is to use the right augmentation scheme. The best accuracy is obtained by cropping and using color distortion. Fine-tuning is applied after the contrastive learning stage to earn top-1 accuracy similar to the supervised model, and in some of the datasets to outperform the supervised model.

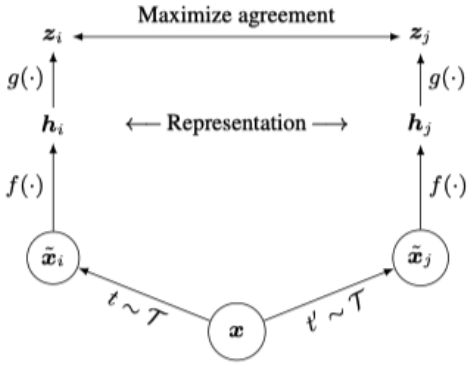


Figure 6: A simple framework for contrastive learning of visual representation
Image from: https://arxiv.org/pdf/2002.05709.pdf page. 2

Training unsupervised with contrastive learning will profit from not having labels. Two different augmentation schemes, *t* and *t'*, are fed through an encoder network, *f(·)*, to get two related representations. The projection head, *g(·)*, and the encoder network are trained by maximising the agreement from the output of the projection head. When training is complete the projection head is discarded. The contrastive loss function will close the gap between the two related representations, while pushing away all other representations.

3.4.2 Supervised Contrastive Learning

The ideas of supervised contrastive learning was introduced in ***Chen***[5], and supplemented in ***Khosla***[6] to outperform cross-entropy. The difference between the supervised and the unsupervised versions of the contrastive model, is the availability of labels in the dataset. The labels is used to create positive pairs to the data points that have the same labels, and negative pairs to the data points which does not have the same labels, as shown below, Figure 7.

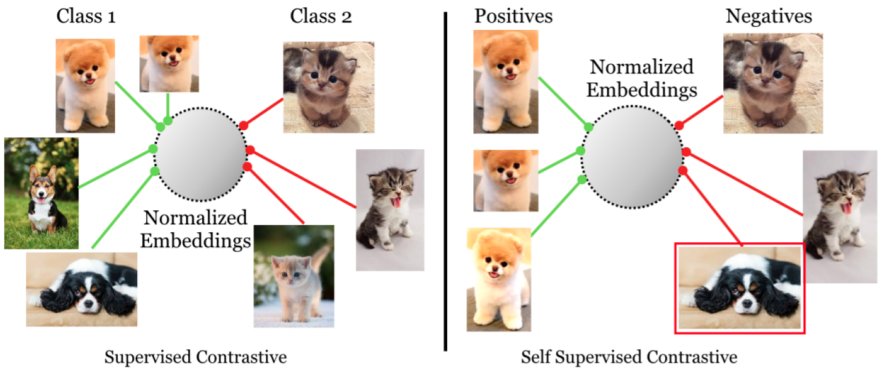


Figure 7: Supervised contrastive vs unsupervised contrastive
Image from: https://arxiv.org/pdf/2004.11362.pdf page. 2

In practice that means that the supervised model will pull images which have the same room label closer together, while pushing away images of rooms with different labels. Like in the unsupervised version, two related representations are created, however multiple possible pairs are present, and not only the one positive pair created from the correlated representation in the unsupervised version.

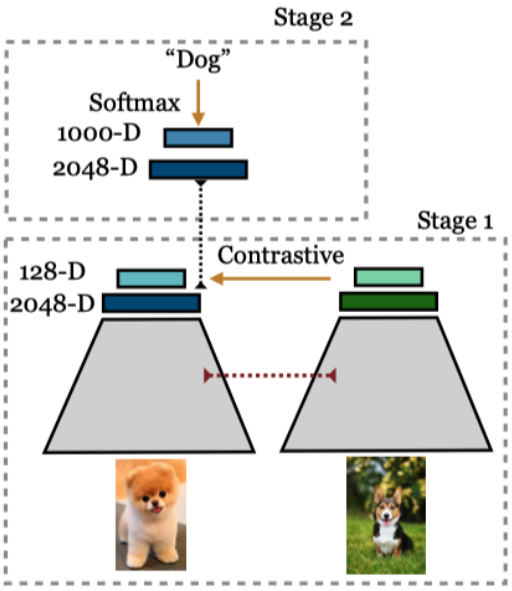


Figure 8: The 2 stages of supervised contrastive learning
Image from: https://arxiv.org/pdf/2004.11362.pdf page. 13

The two stages of classification with supervised contrastive learning, Figure 8. The first stage of the contrastive learning works as described in Figure 7. The second stage of the contrastive classifier is trained with a categorial cross-entropy classifier on top of the encoder network, i.e. without the projection head used when training contrastive.

3.5 Measurement of the Similarity

With respect to the Problem Definition the purpose is to find similarities in rooms. An accuracy measurement is created to calculate the similarity between rooms. For the purpose of this report it is referred to as the top-1 accuracy. The classification accuracy is not considered, because the dataset only has 6 different classes, and is achieving above 95% accuracy with all investigated model setups. The top-1 accuracy is calculated by creating a similarity matrix between all feature spaces retrieved from the CNN, for each of the data points/images accuracy is then evaluated to true/1 if the closest image is of the same class and false/0 if not. The top-1 accuracy is the mean of the calculated 1-D vector calculated from the similarity matrix.

3.6 Background Takeaways

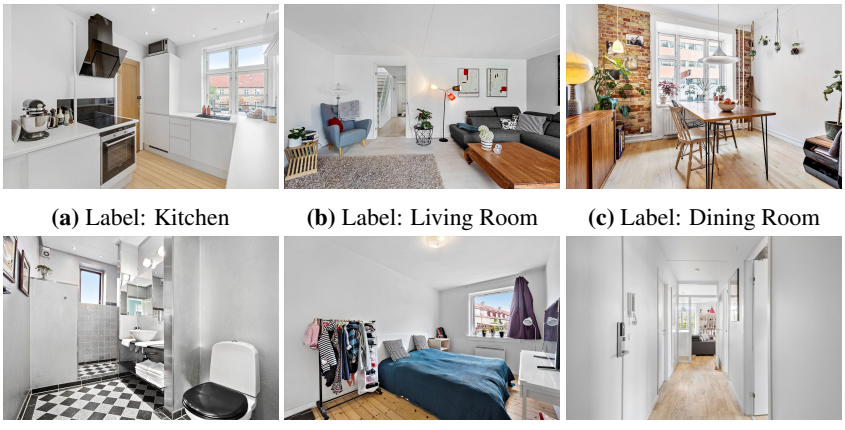
The key takeaways from this section, is that through transfer learning it will be possible to for a general-domain CNN to adapt in-domain like feature representations through limited in-domain data. It is also acknowledged that the authors of ***Khosla***[6], proved to achieve state of the art top-1 accuracies for the ResNet50 model by supervised contrastive learning, when tested on the ImageNet dataset.

4 Experimental Setup

This section provides an overview of the experimental setup. It includes an examination of the limited in-domain dataset, available to study during this project, the ImageNet dataset used to pre-train the general-domain model, and the Places365 dataset used to pre-train the in-domain model,

4.1 Danish Real Estate 2019 Data (DRE19)

This dataset was collected and catalogued by ***Ingwersen***[1] and the author using web crawlers targeting various danish real estate websites. The entire dataset consists of 21000 unique images, however only **6415** have been labeled. For training the remaining part of the dataset, an unsupervised method like the one proposed in subsection 3.4, should be considered for future work.



(a) Label: Kitchen **(b)** Label: Living Room **(c)** Label: Dining Room

(d) Label: Bathroom **(e)** Label: Bedroom **(f)** Label: Entre

Figure 9: Sample images from each class

The labeled dataset consists of 6 classes. The labeled dataset is split into a training, validation and testing set, of respectively: 60% 20% 20%.

	kitchen	living_room	bed_room	bath_room	dining_room	entre	**total**
Train	745	702	622	639	583	541	**3842**
Test	256	231	218	210	195	164	**1281**
Validation	253	239	205	212	194	182	**1292**
Total	**1254**	**1172**	**1045**	**1059**	**972**	**887**	**6415**

Table 1: Labeled images from various Danish real-estate pages

4.2 ImageNet

The general purpose model used for transfer learning, is trained on the `ImageNet`[7] dataset, consistent of more than 14 million hand annotated images belonging to more than 20.000 categories. For the purpose of this project, a subset of the dataset with 1.000 classes has been chosen.

Figure 10: Example of ImageNet data. Image source: [8]

As the pictures show in the above figure, this general purpose model is well trained to detect various different objects.

4.3 Places365

This in-domain dataset consists of 18 million images with 385 scene categories.

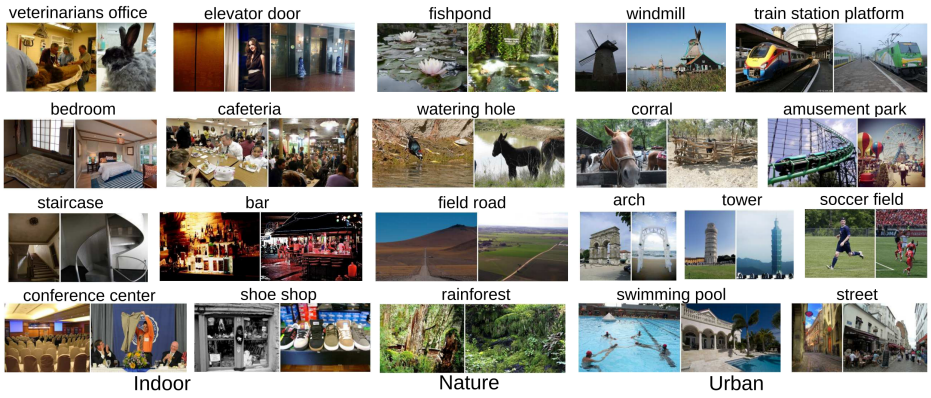


Figure 11: Places macro-classes. Image source: [9]

The dataset also includes outside scenarios and objects, this might be useful if expanding the Danish Real Estate dataset to also include other classes like exterior of homes or balconies.

Figure 12: Places indoor scene categories. Image source: [9]

The above scenarios showcase the strength that the Places365 dataset has against the ImageNet dataset. The categories of the Places365 datasets is very similar to the categories in the Danish Real Estate dataset, and it has already been proven to correctly identify similarities in ***Ingwersen***[1].

5 Methodology

The method used to fulfil the goal introduced in Problem Definition is as follows. The Dataset described in the Danish Real Estate section, is forward propagated through the two encoder networks, to obtain a 2048 and a 1280 dimensional feature vector. The encoder networks used are respectively: Resnet50[10] and EfficientNetB0[11], with

ImageNet Weights as described in ImageNet. During training of the model a dense classification layer, with the 6 room classes and softmax activation is added. After training the model the classification layer is discarded, and the test dataset is used to create feature vectors to measure the top-1 accuracy.

For training the contrastive model, the same 2 encoder networks were chosen and the same ImageNet weights were used. A linear 128 dimensional projection layer were added on top, as described in ***Khosla***[6]. *Khosla* states that the larger the batch sizes the better the model (up to 6000), however due to the limited DRE19 size, only batch sizes of 64 were used in this model. The supervised contrastive model were chosen because it was proven to be state of the when tested on the ImageNet test set.

5.1 Presentation of Frameworks

The main components used to produce results:

- A preprocessing module *Prep(·)* and a augmentation module *Aug(·)*. *Prep(·)* prepares DRE19 to match the inputs required by the models, this means to resize images, and to center the color channels around the mean of the Dataset (ImageNet), used by to pretrain the models. The means for the RGB-values of ImageNet are [103.939, 116.779, 123.68]. *Aug(·)* takes as input *Prep(x)*, where $x = DRE19$, and uses 2 different augmentation schemes, flipping and cropping of the images. These two augmentation schemes were used to create new versions of the images, to get more training data. *Prep(·)* were used for the training, validation and test subsets of DRE19, while *Aug(·)* were used just on the training data set.

- Two encoder networks, EfficientNetB0: *Efn(·)* and Resnet50: *Resnet(·)*. The two networks have both been pre-trained using the ImageNet subset, with 1000 different objects. *Resnet(·)* maps x to a feature vector of size 2048, and the network has 23.6 million parameters. *Efn(·)* maps x to a feature vector of size 1280, and the network has 4 million parameters. The efficiency of EfficientNetB0 is proved to be 1% point more accurate than Resnet50 in [11], when used on the ImageNet test set.

- Contrastive training module *Supcon(·)*, which is a concatenation of either of the networks *Enc(·)* described above and the projection head *Proj(·)*. The projection head is a linear layer with the dimension 128. The training of weights is updated using `max_margin_contrastive_loss(Proj(Enc(x)), labels)`[3], where x is the preprocessed augmented images.

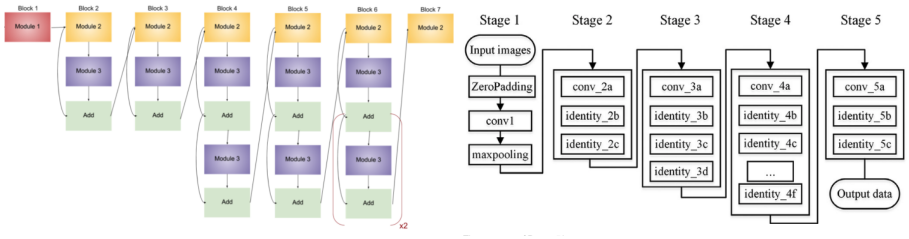
- Similarity module *Sim(·)*, which is used for measuring the top 1-accuracy of the trained network. *Sim(·)* accepts an encoder network *Enc(·)*, and the test subset of DRE19, the part of DRE19 which wasn't used when training the model. The test dataset is forwarded through the *Enc(·)* to obtain feature vectors for all images. A distance matrix between all feature vectors is then created, *dmat(i,j)*, where i is the images, and j is the distances to the j'th element. The diagonal is the case where $i = j = 0$ is true, and is removed. For each images, the lowest distance is identified, and `if label(i) = label(j): count ++`, where

[3] https://raw.githubusercontent.com/wangz10/contrastive_loss/master/losses.py

count is number of true matches. The top-1 accuracy is defined by `top-1acc = count/#images-1`. In practice the top-1 accuracy is an indicator of how well the model is to find similar rooms.

5.2 Transfer Learning

Image a:[4] Image b:[5]



(a) Structure of EfficientNetB0 **(b)** Structure of Resnet50

Throughout the experiments conducted in this study different stages of the encoder networks were fine-tuned. The amount of parameters fine-tuned ranged from the very last layer to more than half of the parameters in both encoders. It was found that the more layers fine-tuned, the better accuracy, up to a certain point. The time consumption also increases, with the amount of parameters fine-tuned. Because this project only focused on being able to obtain in-domain CNN like accuracy for the general domain CNN, there was no goal for being time efficient. The maximum amount of parameters fine-tuned on EfficientNetB0, were $\sim$ 3 millions, out of the total amount of 4 millions, in practice this meant to train the entire stage 7, and block c and d of stage 6. The maximum amount of parameters trained on ResNet50 were $\sim$ 15 millions, out of the total amount of 23.6 millions, in practice this meant to train the entire stage 5. To achieve the best model, early stopping were introduced to measure improvements on the validation dataset. All models in the Results section stopped early, which suggest they all stopped at the optimal point.

[4] https://towardsdatascience.com/complete-architectural-details-of-all-efficientnet-models-5fd5b736142
[5] https://www.researchgate.net/figure/The-structure-of-Resnet50_fig4_331354522

5.3 Contrastive Learning

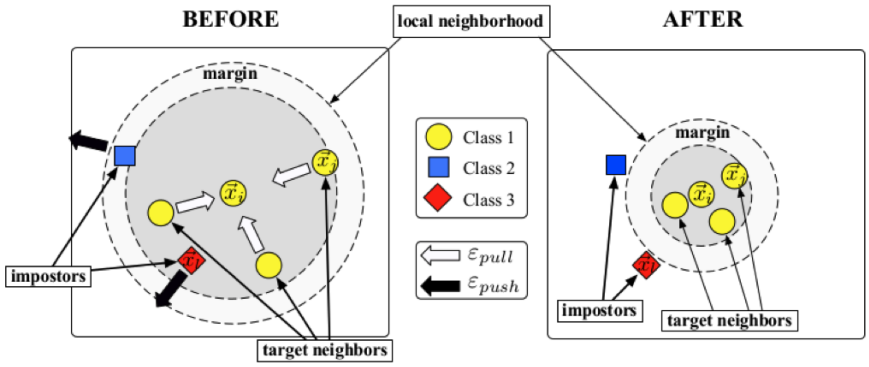


Figure 14: Crontrastive Loss in action[6]

The contrastive loss model implemented in this project, had the intention of binding rooms with similar labels closer together and pushing apart rooms with different labels, as seen in Figure 14, to achieve a higher top-1 accuracy. This was done using the contrastive loss function, showcased above. The loss function takes as input the feature vectors, the image labels and a margin. As shown above, the loss function will minimise the distance between feature vectors of same class, and maximise the distance to feature vectors of different classes for all feature vectors within the margin. For the object of the project, the margin were set to 1. This number was chosen because most of the distances ranged from 0.5 to 5. The implemented method used in ***Khosla***[6] did not use a pre-trained network, unlike the implementation used in this project.

5.4 Previous Work and improvements

A previous study by ***Ingwersen***[1], found that the general-domain model (ResNet50 with ImageNet) did not manage to produce meaningful representations of rooms. ***Ingwersen*** made the choice of focusing on the in-domain model, which was a VGG16 model pre-trained on the Places365 dataset, as this was on the baseline level 10% better than the general domain model. See appendix: A. ***Ingwersen*** reported the baselines of the general-domain model to have a top-1 accuracy of 33% when using ResNet50 preprocessing scheme. The author of this study found a problem when rerunning the baseline tests from ***Ingwersen***, the ResNet50 preprocessing scheme was used incorrectly, and therefore reported a wrong baseline top-1 accuracy for the ResNet50 model. The new improved baseline suggested that the general-domain model would perform almost as good as the in-domain VGG16 model used in ***Ingwersen***. The below table show the baseline improvements:

Model and preprocessing scheme	Improved	***Ingwersen***
Effecientnet	0.719968	**NA**
Resnet50	0.655529	0.322

Table 2: Top-1 accuracy of Baseline models

These baseline results should be compared to the baseline reported in ***Ingwersen***, and yields an improved baseline of >30% points. This improvement suggests that when fine-tuning the general-domain model, the model should be able to compete with the trained in-domain model implemented in ***Ingwersen***. The top performing models in ***Ingwersen*** is shown in appendix: B. The best performing model in ***Ingwersen*** has a top-1 accuracy of **78.3%**, this is called the **Gold Standard** throughout this study, because it is what this project aim to close the gap to.

5.5 Optimizers

In the novel stages of the project different optimizers were tested. The 2 optimizers tested were Adam and SGD. Different learning rates were also tested. The accuracy achieved is as follows:

Optimizer\LR	0.005	0.001	0.0005	0.0001	0.00005
Adam	27%	**33%**	32%	32%	33%
SGD	35,9%	**36%**	35%	35%	NA

Table 3: Top-1 Accuracy, with regards to different learning rates and optimizers.

The SGD is a stochastic gradient descent optimizer, which stochastically updates parameters. SGD proved to work well with transfer learning and the problem definition of this project. Adam were investigated, because it was described in ***Ingwersen***[1] as maybe being the better choice for that project. ***Ingwersen*** trained auto encoders on top of the in-domain model, the teach the model to do better representations, and used Adadelta as the optimizer. However Adadelta is not optimal to use when transfer learning, and SGD were chosen because it outperformed Adam. The accuracies above, were measured before the preprocessing scheme were improved, and was unfortunately not revised after the improvement, why the accuracies are below the baseline results.

5.6 Revising the Dataset

During the experiments, different rooms were identified as being labeled wrong. Therefore the entire dataset were revised. ~ 10-20% of the living rooms and bedrooms were empty, and were removed from the dataset, because these empty rooms will not help home seekers in choosing their preferred aesthetics. This had the effect of binding images with similar labels closer together, appendix: E. The revised dataset is used to train all models in Results.

6 Results

Ingwersen[1] claims that the in-domain model Places365 has a huge advantage over the general-domain model ImageNet. As shown in Previous Work and improvements, this was built on the wrong assumptions, because the baseline accuracies were misleading. The new knowledge obtained about the true baseline accuracies in Previous Work and improvements prepares the ground for achieving the goal described in Problem Definition. It is in this section it is investigated if it is true that the in-domain model does indeed have the huge advantage claimed in previous study by ***Ingwersen***. Or if it is possible to fine tune a general-domain model to perform equally as well as the in-domain model. To do this the general-domain model were trained according to the method described in: Transfer Learning.

6.1 Transfer Learning

6.1.1 ResNet50

Through the early stages of the project, it was really difficult to get any notable improvements to the ResNet50 model. Only small improvements could be seen on the top-1 accuracy, which went from 32% to 35% after training with the revised dataset for 20 hours, after the bug was found in the preprocessing scheme, the top-1 accuracy went from 35% to 65.5%. This seemed to be a huge improvement, and because a proper baseline were found, the actual transfer learning could begin. As suggested in ***Ingwersen*** the preprocessing scheme should have a huge impact on scoring better top-1 accuracies, and especially the places preprocessing scheme should be the better choice, as per: Appendix: A. To get the most accurate results different layers were trained, and what seemed to be most accurate to train were to train the entire stage 5, as per Appendix: C:

Model\Processing Scheme	Places365	ResNet50	EfficientNet
Resnet50	0.522673	0.773270	0.662689

Table 4: Top-1 accuracies for the ResNet50, w.r. to different preprocessing schemes.

In the above table, the best performing top-1 accuracy for the ResNet50 model, with stage 5 fine-tuned, is the version trained on the ResNet50 preprocessing scheme.

6.1.2 EfficientNetB0

The EfficientNet model, which is a more compact and complicated model compared to the ResNet50, have been investigated. This was done because the feature vector it produces is smaller, and it was believed to be able to represent the rooms at least equally as well as the ReNet50 model, because it was proved to performing a better on the ImageNet test set in [11]. To get the best optimal number of layers trained, different layers were investigated. Below is the top-1 accuracy across layers trained:

Trainable layer	Top-1 Accuracy
Entire stage 7	76%
Entire Stage 7, and Stage 6 (D,C)	77,6%
Entire Stage 7 and entire Stage 6	71%

Table 5: EfficientNetB0 trained with different with different layers.

Different preprocessing schemes were also investigated, to see how well each performed on this model:

Model\Processing Scheme	Places	Resnet50	EfficientNet
EffecientNet	0.735879	0.731901	0.776452

Table 6: EfficientNetB0 trained with different preprocessing schemes.

In the table above, the best performing top-1 accuracy for the EffecientNet model, with the entire stage 7 and stage 6 (D,C) fine-tuned, is the version trained on the EfficientNet preprocessing scheme.

6.1.3 Comparison to Gold Standard

The above results should be compared to the Gold Standard. ***Ingwersen*** managed to achieve a top-1 accuracy of 78.3%, which was a 2.3% increase compared to baseline model in his study. The best model in ***Ingwersen*** were obtained by mapping the in-domain model to a 512 dimensional vector. The gap between the baselines obtained in this study and the Gold Standard:

Model	Baseline top-1 Accuracy	Gap to Gold standard
EffecientNet	0.719968	6.3% points
ResNet50	0.655529	12.3% points

Table 7: Top-1 accuracy of Baseline, and their gap to the Gold Standard.

The best performing transfer learned model for each general purpose CNN is reported below, together with their gap to the Gold Standard:

Model	Top-1 Accuracy	Gap to Gold standard	Improvement
Effecientnet	0.776452	0.7% points	5.6% points
Resnet50	0.773270	1% points	11.3% points

Table 8: Top-1 accuracy of the best performing transfer learned models, and their gap to the Gold Standard, the improvement from the baseline is also reported.

The best performing transfer learned ResNet50 model had a large gap to the **Gold Standard**, through transfer learning the gap was almost closed, however if this result is compared to in-domain model with the same feature vector dimension, the two models perform equal. The in-domain 2048 model also have a top-1 accuracy of 77.3% as reported in Appendix: B. The best performing transfer learned EfficientNetB0 model had

a smaller gap to the **Gold Standard** and almost closed the gap, 0.7% off. It should be taken into account that the feature vector of the EfficientNetB0 model had dimensions of 1280 while **Gold Standard** had dimensions of 512.

6.2 Contrastive Learning

Contrastive learning was applied to the general-domain model, as an attempt to close the gap even further. The reference code, which the contrastive model of this project were built on is located at: https://github.com/wangz10/contrastive_loss, however due to the different experiment setup, i.e. small batch size compared to ***Khosla***[6], and because the goal of this study was to close the gap between an in-domain model and the general-domain model, the contrastive learning needed to be applied to the pre-trained general-domain network, which is not the case in ***Khosla***, ***Khosla*** trained their network from the bottom and up, this was not possible in this study, due to the limited dataset. This led to problems, which caused the loss to update all weights in the network to "NaN"s halfway through training. Below is the loss measurements, which show the loss being minimised for the first 30 epochs, after this no loss could be reported.

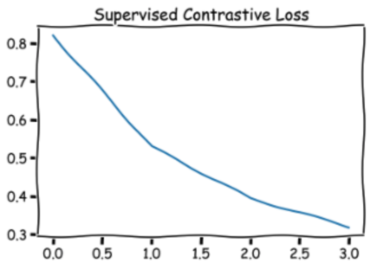
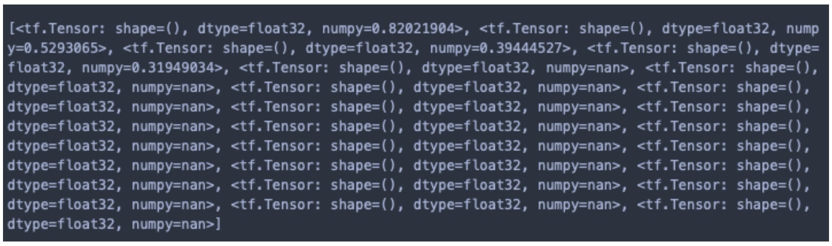


Figure 15: The loss of the network seems to be minimised for the first 30 epochs. After this no loss were reported, and all the weights were changed to "NaN".

The screenshot of the reported loss value are reported below:



```
[<tf.Tensor: shape=(), dtype=float32, numpy=0.82021904>, <tf.Tensor: shape=(), dtype=float32, nump
y=0.5293065>, <tf.Tensor: shape=(), dtype=float32, numpy=0.39444527>, <tf.Tensor: shape=(), dtype=
float32, numpy=0.31949034>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>, <tf.Tensor: shape=(), dtype=float32, numpy=nan>, <tf.Tensor: shape=(),
dtype=float32, numpy=nan>]
```

Figure 16: The first 30 epochs seems to work fine. But then "NaN" error occurs.

Unfortunately it was not possible to recover the weights before the model broke, and no top-1 accuracy could be measured. However as the loss measurements show, it seems to be working, and progressing well, which suggest that the idea might be worth to implement. The failure of the contrastive learned model is further described in Conclusion.

6.3 Show Case: How well does the models predict similar rooms?

The plots below show how well the two general domain models are able to produce meaningful representations of features in rooms. For full size image plots go to Appendix: D.

6.3.1 ResNet50

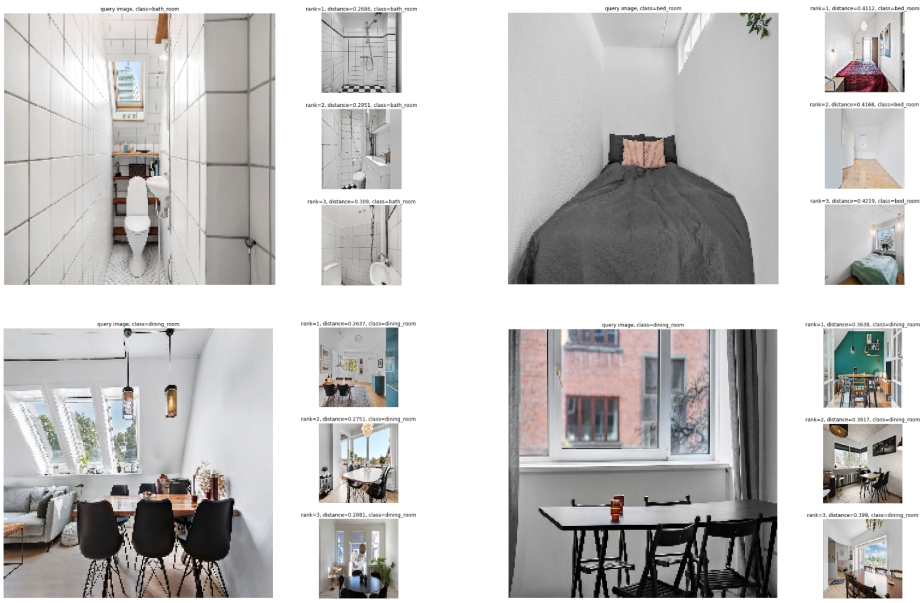


Figure 17: ResNet50, prediction of the 3 most similar rooms

The ResNet50 model have a top-1 accuracy of 77.3%, and are able to capture key features in rooms as seen in the above plotted images. The first case, the model is able to capture the distinctive tiles which decorates the walls of the bathroom. In the second case, bedroom, the model is able to capture the size of the room. Home seekers who are looking for homes, which do not use excessive space on the bedrooms, will be able to mark this as a preferred aesthetic preference to get homes that matches that criterion. Same thing goes for the tiles in the bathroom.

6.3.2 EffecientNet

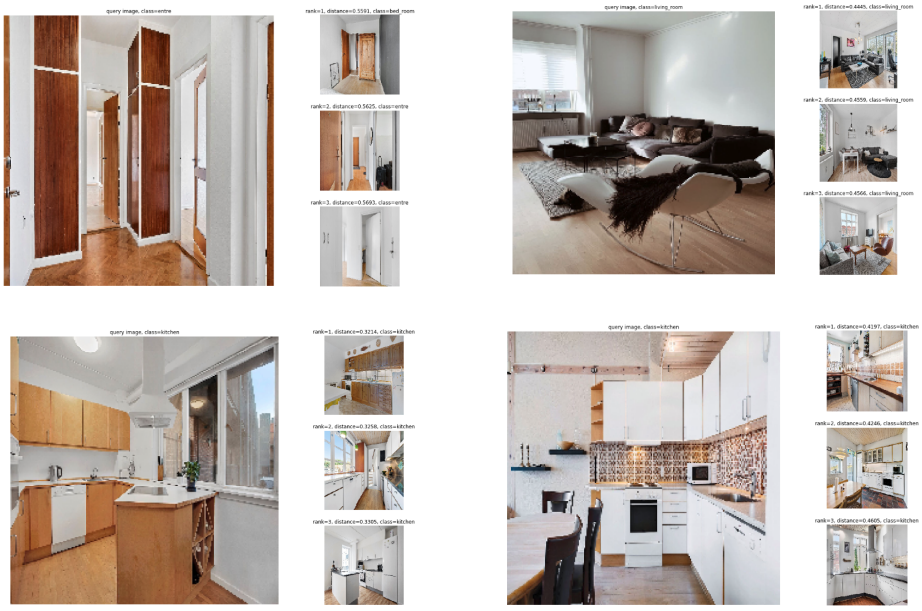


Figure 18: EfficientNetB0, prediction of the 3 most similar rooms

The EffecientNet model have a top-1 accuracy of 77.6%, and are also able to produce meaningful representations of the rooms. These images, show a few interesting results as well. The first case, a bedroom is matched as being the nearest room to the entre, and by looking at the images, it is clear that the model has caught the use of wooden materials. The first kitchen case also showcase the ability to capture wooden materials used in the kitchen, while the second kitchen case has caught the use of retro tiles on the walls and floor of the kitchens.

6.3.3 Untrained ResNet50

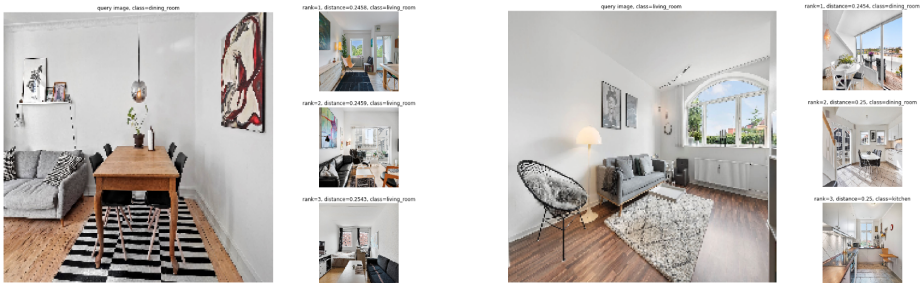


Figure 19: The untrained model, not capturing alot of the key features, like rugs and windows. Also predicts wrong room label.

Above is the untrained model showcased, which have a top-1 accuracy of 65.6%, to establish if the transfer learning has proved to be a success. The images tells the story that the untrained model is not performing equally as well as the transfer learned models, on the sole argument that it is not capturing the right labels for the classes. The model also does not capture the key features of the rooms, like the lack of windows in the first living room image, and also not venetian style window in the second living room image.

7 Conclusion

Machine learning is a fast developing field of science. Everyday new frameworks and projects become available to the public through open source libraries, like the ones used in this study (*Tensorflow and Keras*). The reason for the fast developing field is that the world we are living in today is a data driven world, and all big companies harvest data like never before. But for small companies it might not be possible to harvest the same amount of data due to lack of resources. This study has investigated different ideas of how small companies like the real estate agencies in Denmark might be able to grow through open sourced libraries and limited datasets.

In this study it has been established that by transfer learned training it was possible to use a general-domain CNN as an accurate feature extractor, for which real estate agencies can increase customer engagement. The pretrained models were able to learn meaningful representations of rooms, through training with limited datasets. It was proven that the general-domain model could be trained to obtain an in-domain like top-1 accuracy for the DRE19 dataset. Through investigation of the images it was experienced that the models could be trained to learn meaningful feature vectors of the rooms, which directly could be translated as the model being able to learn visual aesthetics of rooms.

A previous study ***Ingwersen***[1] claims that the preprocessing scheme had an important factor in producing meaningful representations. In this study however that claim were refuted. The preprocessing scheme should be chosen according to how the models were pre-trained.

Investigation of the contrastive learning have been done and also tried implemented, the early stages of the training seemed to work well, but the model broke halfway through, and no results were acquired. The author of this paper believes that it would have proven to be a success if the training were completed. The author also believes that the model failed to complete training due to the limited dataset size, and small batch sizes. The contrastive loss function might have had a batch were there were just one of any of the classes, and then were unable to minimise the loss to an other case, beacuse no other case could be found. However this should be further investigated.

In addressing the Problem Definition of this study, it proved to be possible to close the gap between the general-domain models and the in-domain model. The results suggest that reducing the dimensionality of the feature vector could have a saying when it comes to the top-1 accuracy used in this report. This was observed in this study, through the EffecientNetB0 model performing better than the ResNet50 model, and in

previous study by ***Ingwersen***[1].

7.1 Future Work

7.1.1 Semi-Supervised Training on Remaining DRE19 Data

As described in DRE19 only approximately a third of the full dataset has been labeled. As contrastive learning have been investigated, it would be a possible solution to use the unsupervised version of contrastive learning, as described in **textitChen**[5], to label the remaining part of the dataset. As the real estate market never cease to exist, it is crucial for the real estate agents to implement a scaleable solution to deal with the never ending flow of new homes entering the market. The implementation of unsupervised labeling of images, together with a feature extractor similar to the one proposed in this model, will give real estate agencies an advantage over other real estate agencies on the market, by increasing customer engagement through new improved ways of searching for homes, that match the home seekers own preferences. Gaining a larger labeled dataset might also be key to achieving the larger batch sizes, that are required for the supervised version of contrastive learning described in ***Khosla***[6] to be implemented.

7.1.2 Supervised Contrastive Learning

Further studying of the supervised contrastive learning should be done in order to conquer a higher top-1 accuracy. An easy implementation, to see if the authors idea about the too small batch sizes, would be to try and run the same training, but with increased batch sizes. Maybe a batch size should be equal to the entire dataset.

Appendix

Appendix A: Baseline tests from *Ingwersen*[1]

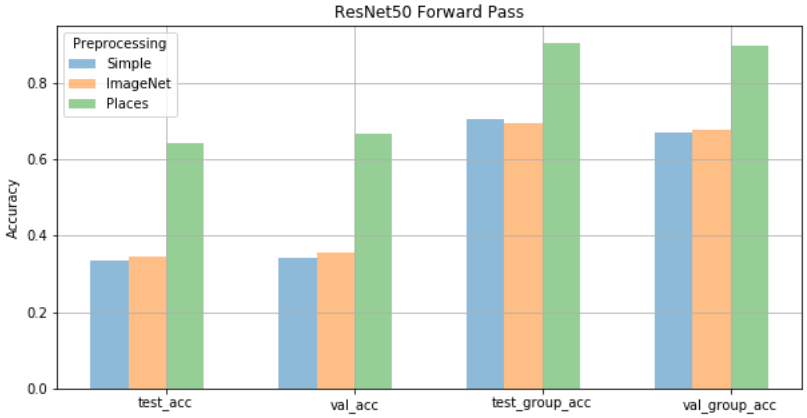


Figure 20: Baseline performance of ResNet50

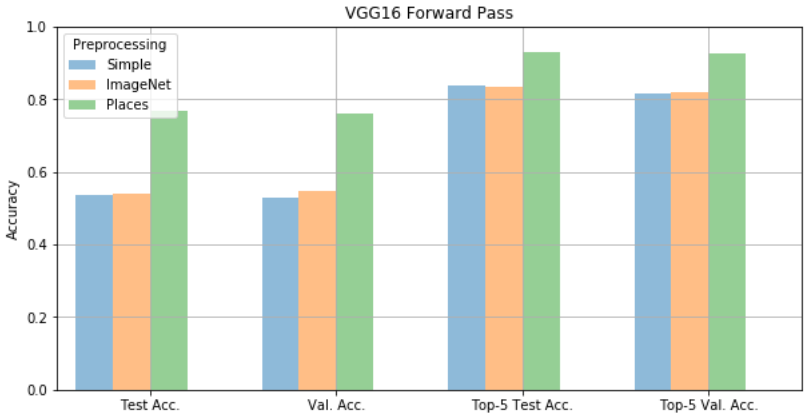


Figure 21: Baseline performance of VGG16

Model	Preprocessing	Test Acc.	Val. Acc.	Top-5 Test Acc.	Top-5 Val. Acc.
ResNet50 (maxpool)	Places	0.6438	0.666	0.904	0.898
VGG16 (fc_1)	Places	**0.769**	**0.760**	**0.929**	**0.927**

Appendix B: Top performing models in *Ingwersen*[1]

Pretrained	Architecture	Test Acc.	Val. Acc.	Top-5 Test Acc.	Top-5 Val. Acc.
VGG16(fc_1)	4096, 2048	**0.782 (+ .013)**	0.773 (+ .013)	0.937 (+ .008)	0.944 (+ .017)
VGG16(fc_1)	4096, 3072	0.778 (+ .009)	0.765 (+ .005)	0.943 (+ .014)	0.940 (+ .013)
VGG16(fc_1)	4096, 512	0.771 (+ .002)	**0.783 (+ .023)**	**0.945 (+ .016)**	**0.951 (+ .024)**
VGG16(fc_1)	4096, 128	0.775 (+ .006)	0.774 (+ .014)	0.936 (+ .007)	0.944 (+ .017)

Table 9: Experimentation with AEs (Places preprocessing), parentheses indicate improvement over baseline

Appendix C: Training different layers of ResNet50

Trainable layer	SGD
Stage 5, block B and C	35.1%
Stage 5, block A, B and C	36%
Stage 5 and block C of stage 4	32%

These tests, were created before noticing the problem with the preprocessing function, introduced in ***Ingwersen***[1].

Appendix D: Similarity plots

ResNet50

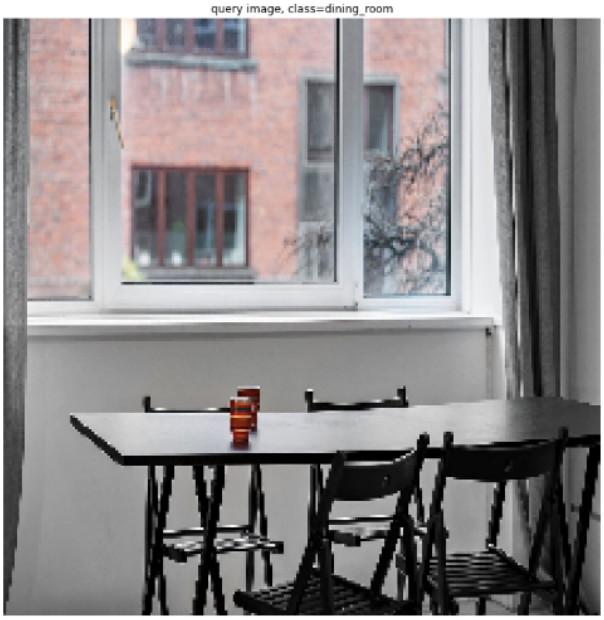
query image, class=dining_room

rank=1, distance=0.2637, class=dining_room

rank=2, distance=0.2751, class=dining_room

rank=3, distance=0.2881, class=dining_room

query image, class=dining_room

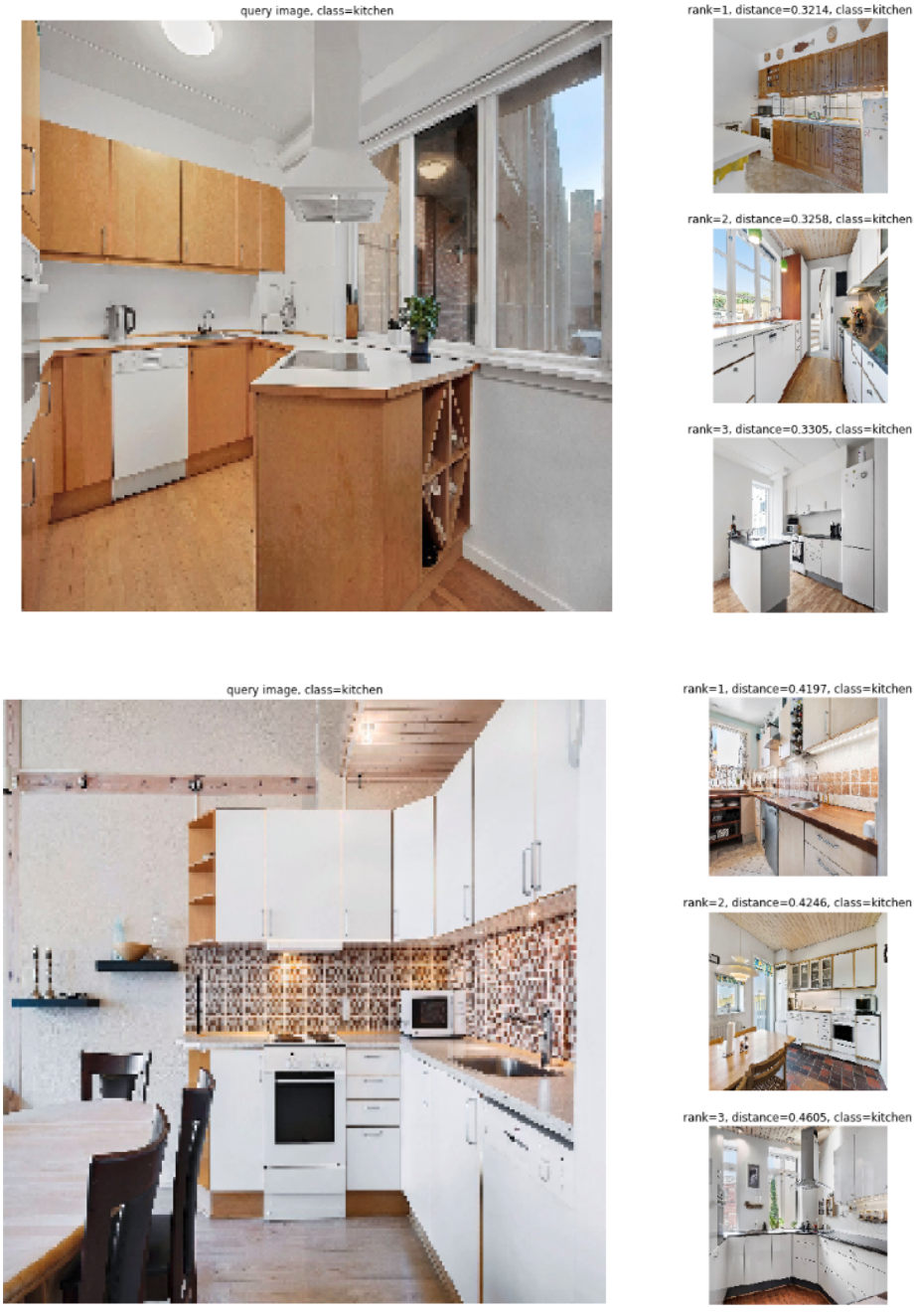


rank=1, distance=0.3638, class=dining_room

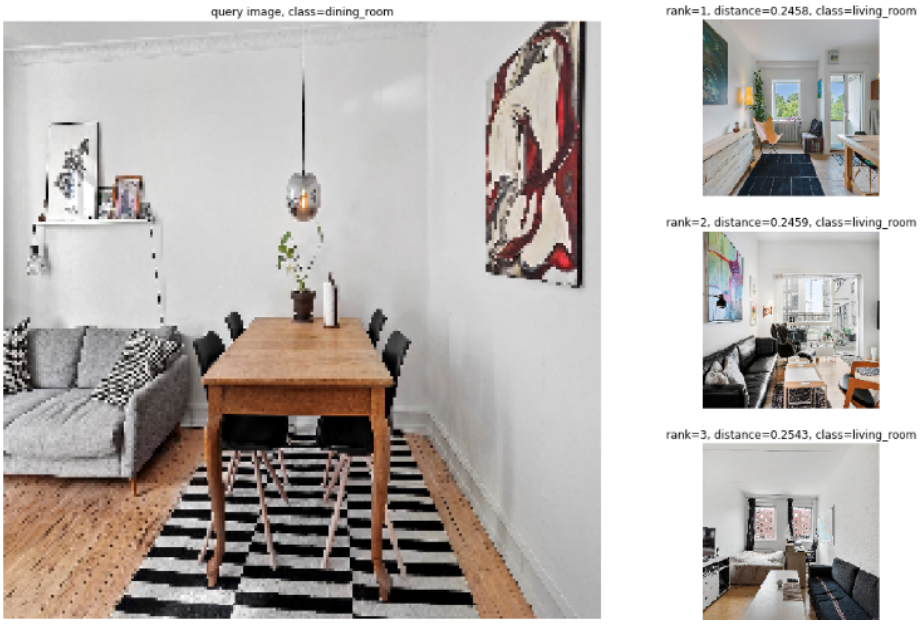
rank=2, distance=0.3817, class=dining_room

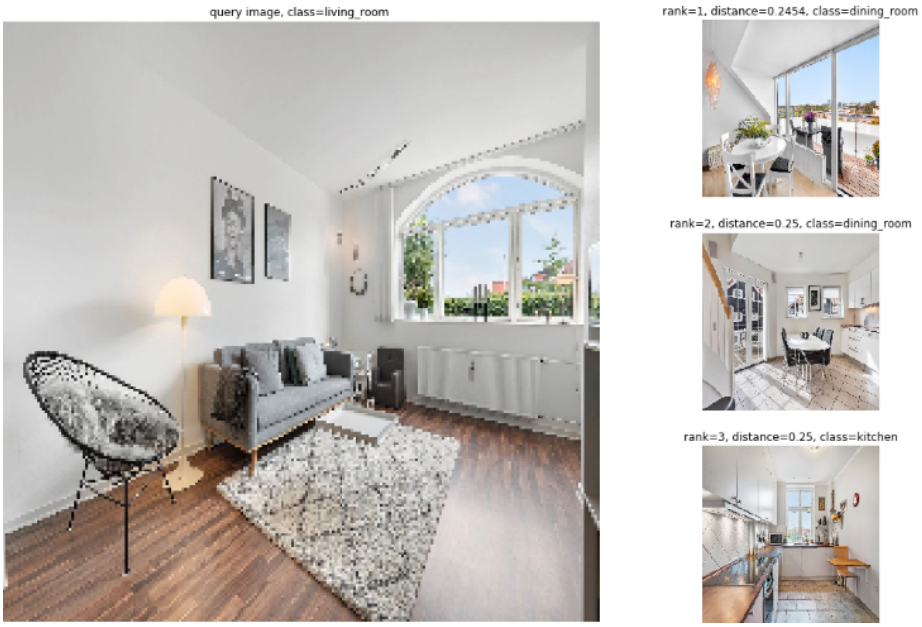
rank=3, distance=0.399, class=dining_room

EfficientNetB0


query image, class=kitchen
rank=1, distance=0.3214, class=kitchen
rank=2, distance=0.3258, class=kitchen
rank=3, distance=0.3305, class=kitchen
query image, class=kitchen
rank=1, distance=0.4197, class=kitchen
rank=2, distance=0.4246, class=kitchen
rank=3, distance=0.4605, class=kitchen

The Untrained Model





Revised Dataset, compared to old dataset

	Revised Dataset	Dataset
bath_room	0.453	0.668
bed_room	0.390	0.660
dining_room	0.321	0.556
entre	0.321	0.535
kitchen	0.496	0.747
living_room	0.385	0.754
Mean:	0.394	0.653

Table 10: Mean distance based on room labels

Bibliography

[1] Adam Frederik Ingwersen Linnemann. "Transfer-Learned Autoencoders for Visual Similarity". In: (2020). URL: https://github.com/adamingwersen/dre19_report/blob/master/master.pdf.

[2] Vincent Dumoulin and Francesco Visin. "A guide to convolution arithmetic for deep learning". In: (2018). arXiv: 1603.07285 [stat.ML].

[3] Karthik E. *A Framework for Fast Scalable BNN Inference using Googlenet and Transfer Learning*. 2021. arXiv: 2101.00793 [cs.CV].

[4] Sinno Jialin Pan and Qiang Yang. "A Survey on Transfer Learning". In: *IEEE Trans. on Knowl. and Data Eng.* 22.10 (Oct. 2010), pp. 1345–1359. ISSN: 1041-4347. DOI: 10.1109/TKDE.2009.191. URL: https://doi.org/10.1109/TKDE.2009.191.

[5] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *arXiv preprint arXiv:2002.05709* (2020).

[6] Prannay Khosla et al. "Supervised Contrastive Learning". In: (2020). arXiv: 2004.11362 [cs.LG].

[7] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[8] Khawaja Ahmed, Aun Irtaza, and Muhammad Iqbal. "Fusion of local and global features for effective image extraction". In: *Applied Intelligence* (Apr. 2017). DOI: 10.1007/s10489-017-0916-1.

[9] Bolei Zhou et al. "Places: An Image Database for Deep Scene Understanding". In: *Journal of Vision* 17.10 (2017), p. 296. ISSN: 1534-7362. DOI: 10.1167/17.10.296. arXiv: 1610.02055.

[10] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[11] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: (2020). arXiv: 1905.11946 [cs.LG].