“Con fundamento en el artículo 21 y 27 de la ley federal del derecho de autor y como titular de los derechos moral y patrimonial de la obra titulada “COMPACIDAD, UN BREVIARIO AL INFINITO”, otorgo de manera gratuita y permanente al instituto tecnológico autónomo de México y a la biblioteca Raúl Baillères Jr. Autorización para que fijen la obra en cualquier medio, incluido el electrónico y la divulguen entre sus usuarios, profesores, estudiantes o terceras personas, sin que pueda percibir por la divulgación una contraprestación”

Amaury Gutiérrez Acosta

FECHA

FIRMA

A la ε-cercanía.

- A Lety por plantar la semilla de lo que soy.
- A Ricardo por regar esa semilla, no sería nada sin ti.

Don't Panic

Contents

Preface

It is a very exiting moment for the field of computer vision, machines are now capable of performing task that we thought were impossible reaching new milestones each year. It has been a long time since computers were just huge furniture in cold university rooms. Computer power has grown exponentially since those days. Lately techiniques that where discarded because the were computation intense are now being unburried and have been showing great results in today machines.

I wanted to explore the possibilities that these techniques can offer in classic problems such as landcover classification. In particualar, I wanted to teach a neural network how to recognize a specific element in aereal imagery and then use the tailored features as input to train classic classifiers such as random forest or support vector machines.

This work explores the use of Convolutional Neural Networks in the context of a classical remote sensing problem called landcover classification.

This preface serves as an introduction to this work. It gives a short review on the contents of each chapter, and shows how is the thesis structured.

Motivations are explored in chapter 1. We expose why our work is important and we give a clear explanation about the objective of the experiment. Additionally, we mention the scope of the project.

An extensive literature review is shown in chapter 2. All the way back to the famous tecnique to analyse handwritten digits with the convolutional architecture that started the revolution. A review of more modern applications of the techique in more complex situations such as object recognition in images. We explore other experiments that show different uses of the CNNs in remote sensing. Damage assesment in the af-

termath of natural disasters is also explored as it was the main motivation for this study.

Chapter 3 unveils the mathematical details that make this technique to work. The training process using backpropagation is explained. Last layer activations functions such as the sigmoid and softmax are inspected. A brief summary of a convulutional neural network techniques such as the ReLU activation, pooling, and data augmentation is given.

The architecture of our pipeline is explained in chapter 4. This includes the data gathering, data curation, the training of the network and the prediction. Details of the data base are given, including the reasoning behind some decisions. This chapter also explores how did we obtained the images.

The results of the experiment are exposed in chapter 5. Accuracy of the classifier, as well as the validation protocol are clearly exposed here. We also show the final classification map and talk about how it was obtained.

Finally, in chapter 6, we talk about future work, and what are the main conclusions that this work led us to. We include several improvements that can be made to the process in order to obtain better results.

This work was the result of an intership spent on the Stevens Insitute of Technology in Hoboken, New Jersey, during the summer of 2017. It was done under the supervision of Andrea Garca Tapia and Jos Emmanuel Ramirez Marquez.

Chapter 1

Motivation

> Good work is not done by 'humble' men. (...) A man's first duty, a young man's at any rate, is to be ambitious.
>
> A Mathematician's Apology
> G. H. Hardy

In this chapter, we explore the motivation, objective and scope of the present work.

1.1 Motivation

In the National Commission for the Knowledge and Use of Biodiversity we use land-cover maps to analyze and assess the evolution of the environment through time. We make this possible by leveraging classic classification algorithms and a large amount of computing power. While our efforts have been quite productive, these algorithms have certain limits, they rely on the use of the light spectrum. As a consequence, any two categories with similar spectrum footprint will, in all likelihood, confuse the classifier. Curiously enough, humans have little problem distinguish between some of these pairs of categories. For example, crops and grasslands might seem identical to a supervised classifier, but we can certainly tell the difference from one another. This is caused because our brains are not seeing particular pixels and trying to classify them one by one. Instead, our brains look at the whole picture, we focus on zones of the image an all of the information included in them, in other words, we care about context. Con-

volutional Neural Networks take this into account. Each neuron of the network cares only about a certain zone of the image when information flows through the layers of the network, there are certain neurons that activate upon certain stimuli. Taking context into account lets the network to recognize certain features that would be invisible to a classic classifier, for example: shapes and geometries. When we ask ourselves why it is so easy to differentiate crops from grasslands geometry comes as a natural answer. Crops have very particular shapes.

The final objective is to build a comprehensive biodiversity monitoring system. It can be though as two independent efforts. One of these attemps is the Monitoring Activity Data for the Mexican REDD+ program (MADMex) [10] which pretends to monitor the behavior of forest and vegetation across the country by processing satelitte imagery. Another effort is the Mexican National Biodiversity and Ecosystem Degradation Monitoring System (SNMB) [9] which gathers information about species in the different ecosystems that exist in Mexico.

Segnet papers: [3] in [13] they enhace their approach by extending their own architecture to include a Bayesian approach. The idea is to add a model of uncertainty to the CNN and use this information to get more acurrate guesses on each of the pixels. They report that this feature adds some improvement in the level of accuracy for many types of architectures not only their own SegNet.

A wonderful introduction to neural networks in the context of remote sensing can be found in [4].

In the context of natural disasters other options have been considered [15].

DeCAF paper talks about using the features extracted from a neural convolutional network to use in traditional methods. [8]

This is another attempt that adds to the evidence that features engineered by the Neural Network work pretty good off the shelf. [20]

Transfer learning is explored by Yosinski *et al.* [26]. They propose to use an already trained architecture in new tasks by replacing diferent layers and retraining.

In [17] they use the features extracted from the CNN to segment an image.

The possibility of having a single model that can perfom correctly in many different tasks is explored in [12]

A survey on disctint aspects of how transfer learning is used can be found in [19].

The use of active learning together with semisupervised learning tenchniques is explored in [27].

Need to take a look into [5]

Everardo suggested to look into the U net [22].
And the books: [4], [21], [24] ,[11]

1.2 Objective

It is a wondeful time to explore the use of novel technologies in all types of context. It is in human nature to create tools that revolutionize the way it modifies its envirmonment. We want to explore the use of Convolutional Neural Networks in the context of landcover classification. We believe that this field is very promising and will bloom in the next years. By appliying this methods we hope to obtain results that match the ones that we are already getting with classic methods. In Mexico, the National Institute of Statistics and Geography has been in charge of producing periodical maps about urban settlements in the country. However, it is hard to keep track of irregular settlements and it is often prohibitely costly to create these maps from on-site visits. We want to offer a cheap and innovative alternative to detect and index urban areas using the inexpensive satellite imagery that is available.

1.3 Scope

It would be far too ambitious to cover every topic that is involved in the process of the classification using CNNs. We want to explain how do the networks work to a

certain extent, but it is not in the scope of this work to untangle every single detail. In the same fashion, the field of Remote Sensing is far too big to be explored in this work. We assume a certain degree of knowledge in related topics and we expose some mathematical details in the appendix. As we already mentioned, the field of Computer Vision is in its climax. Reviewing every single article that has been writen would be a daunting task. We offer a brief literature review that gives some context about the state-of-the art and we hope to build upon ideas and efforts done by a miryad of people. We aimed to built a system capable of procesing imagery and that could be deployed into a cluster for efficient computation. We had in mind to offer a wall-to-wall product of the Mexican landscape.

Chapter 2

Nothing new under the sun, Reinventing the wheel

> If I have seen further it is by
> standing on the shoulders of giants.
>
> Letter from Sir Isaac Newton to
> Robert Hooke

In this chapter, we talk about the state of the art in computer vision and how it has been used for remote sensing problems. We also give a brief account of natural disaster assessment, and how are these machine techniques applied in this sense. We use Sandy Hurricane as a study case because of the data that was publicly made available by the NOAA.In this chapter the mathematical details of the CNN are exposed. How does backpropagation works, and differences between multiclass and multilabel classification.

2.1 Computer vision

Le Cun *et al.* [6] propose to use an architecture of a multilayer neural network that was able to learn directly from the data with no prior feature extraction. In contrast to the usual path that was used in the context of pattern classification, they created an architecture that was able to automatically extract the features directly from the date without prior manipulation. Instead of using a fully connected network, they proposed a locally connected net. It was capable of extracting local features and passed them down to the subsequent layers in what they called a *feature map*. Each unit took the

information of a 5×5 neighborhood of the pixel in the previous layer. The last layer of the architecture consisted of ten units that represented each of the possible digits. This architecture was trained using backpropagation are now known as Convolutional Neural Networks (CNNs). The big leap forward of their result was that their architecture needed very little information about the task it was performing, they were able to extend the use of their method to other symbols, however, they state that the method was not able to be applied to very complex objects.

Before CNNs became widely used in computer vision tasks, the use of neural networks to automatically detect roads was already being explored [18]. They proposed a simple architecture with a single hidden. They use the now standard procedure of cropping small patches from the complete image and worked on the RGB color space. Aerial imagery in addition to vector information on roads is used as training data. Instead of manually tag the images they proposed a model that asumes certain amount of thickness in the roads, which are tipicaly with no dimension. To reduce the input dimensionality, they applied Principal Component Analysis keeping the most informative principal components. As a method of post-processing, they use a CNN to reduce the noise in the images. Effectively getting rid of false positives and false negatives by using context information. Of the important lessons learnt by this experiment is the value of the random rotations in the training data. It is common that road networks in cities form grids, they realised that a model trained with information from a particular city will perform poorly if data from a new city is shown to it. They found that this can be relived if random orientations are applied to data as when we are dealing with birdeye sight, there is no correct orientation for images. They also found that adding pre-procesing such as edge detection techniques showed no improvement to the performance of their pipeline. This was attributed to the fact that the neural network crafts features of this nature when learning to perform the task of recognizing roads.

There have been several attempts to use features extracted from a CNN to be used in a different context. Michael Xie *et al.* [25] examine this approach by training a CNN on top of the well-known `VGG F` model. First, they replace the fully connected layers on top of that model with a convolutional layer. Then they re-train the features the model learned from ImageNet with aerial images and nighttime images gather from the National Oceanic and Atmospheric Administration (NOAA). While they get nice accuracy results from using daytime images to predict nighttime light, it was not the

purpose of their research. Instead, features crafted by the network are extracted and used to train a model to predict poverty from satellite imagery. In order to do so, they use these features as input for a logistic regression classifier. To compare their model, they train four other models extracting features from a survey, features from ImageNet itself, features from the nighttime light intensities and features from ImageNet and nighttime light intensities at the same time. The transfer model outperforms every model except for the one based on survey data. This strongly suggests that the transfer learning technique is actually extracting complex information from the aerial scenes. They mention that this approach can be useful when conducting surveys is prohibitively expensive.

With the tremendous advances that computer power has suffered in the late years, this has been proven to be incorrect. In 2009 a big image database was gathered and published [7]. Ever since this database became the defacto dataset to test classification methods. A few years later, in 2012 Krizhevsky *et al.* [14] proposed the use of CNNs in this daunting task.

2.2 Remote sensing

In late years groundbreaking advances in computer vision have led to tremendous advances in other science fields. In particular, we are interested in landcover classification.

The use of CNNs in the context of landcover classification was explored by Kussul *et al.* [16]. They used an ensemble of CNNs to obtain state of the art results in the classification of different types of crops using multitemporal and multisensor satellite data. They explore 2 approaches, first they use a 1-D CNN to perform the convolutions in the spectral domain by stacking the different bands from the Sentinel-1 A and Landsat-8 scenes. This process outputs a pixel-wise classification, then they perform a traditional 2-D CNN on the scenes. In order not to lose resolution with the 2-D CNN, they use a sliding window approach assigning the class to the center pixel of the sliding window. Finally, they ensemble both opinions and filter the result to improve the quality of the map.

The usual approach with landcover classification is the use of classical classification methods such as support vector machines (SVM) and random forests (RF). In order

to improve the performance, features must be handcrafted from the original bands. In [23], Grant *et al.* explore the use of Transfer Learning and Data Augmentation in the context of remote sensing images. By exploring well-known high-resolution datasets, they obtain state of the art results.

Transfer Learning (TF) is the process of using an already trained CNN, to

2.2.1 Data augmentation

Data augmentation is a technique used to artificially increment the size of the training dataset by applying an affine transformation to the images. It is often used when tagged data is scarce and difficult to obtain. The usual transformations include rotations and reflections. When using this technique we should be careful about the orientation of the objects, for example, a building upside down makes no sense, so there is no use to make the network learn features on objects that it won't see in the wild. Fortunately, aerial imagery doesn't present this problem. There is no particular orientation that can be considered correct when the pictures are taken from above. This means that we can dramatically boost the size of our dataset.

The reasoning behind this idea is that when we see a picture, our brain automatically orients it into its correct position. By showing the network with different positions and orientations of an object we enrich its knowledge about it.

We can think of the neural network as a newborn kid, in the beginning, they experience its environment for the first time.

2.3 Damage Assessment

2.4 TensorFlow

TensorFlow is a machine learning system developed by the Google Brain Team to supersede its first-generation system, DistBelief. It was built on top of the lessons learned during DistBelief development. One of the fundamental concepts that lead the team to create a new system from scratch was the need for flexibility. TensorFlow was thought

as a way to expressing machine learning algorithms using a common interface with implementations targeting a wide range of devices. Complex models that where first implemented in DistBelief such as Inception got a performance boost of a factor of x6 [1] when it was ported to the new system. TensorFlow was opened sourced on November 9, 2015 under the Apache 2.0 license.

In this section, we will untangle some of the details of how TensorFlow and its graph model work [2].

It uses an elegant data-flow system in which both the operations and the state of the algorithm are represented as nodes and edges in a directed graph. This lets the system to pre-calculate an optimal sub-graph before starting the calculations.

With flexibility in mind, this system lets the user define new operations and register them to use within the framework. Additionally, it was developed to target different platforms, from machine clusters to mobile devices, these implementations are known as *kernels*. Using the same programming model, TensorFlow decides in runtime which pieces to use. This is useful when you take into account the whole development process of a data product and how it evolves. We can think of a common scenario, first, the developer experiments with data in a single computer before deploying the system to train with a larger data set in a cluster of computers, when the model is trained, it can be deployed to an online service which will run on a single computer or it can be implemented to be used in a mobile device for offline use. In each of these steps the underlying environment is completely different, however, TensorFlow adapts automatically to each situation. That's where TensorFlow shines.

As a common interchange data format, it uses tensors. With machine learning algorithms, it is often the case to have sparse data, encoding it as dense tensors is a clever way to save space. As we mentioned before, TensorFlow uses a graph to represent both the state and the operations. Nodes represent operations. Edges represent inputs and outputs between these operations. The system takes its name from the tensors flowing through this pipes. Although it is not of particular importance to our experiment, it is worth mentioning that TensorFlow supports algorithms with conditional and iterative control flow, which means that it can be used, without further tuning, to train Recur-

rent Neural Networks which are very important in fields like speech recognition and language modeling.

The system also provides a library that allows symbolic differentiation. As many machine learning techniques rely on Stochastic Gradient Descent to train a set of parameters, this feature makes easier to explore new techniques as the backpropagation code is automatically produced for any combination of operation nodes.

TensorFlow was built with huge data-sets in mind. It provides an intern library that allows the distribution of datasets that would be to large to fit in RAM. Instead, data can be sliced, taking advantage of how some algorithms work. Additionally, comunication between nodes use lossy compression, taking advantage of the fact that some of the machine learning algorithms are tolerant to reduced precision arithmetic. It is important to mention that it is possible to extract state and information from any particular node in the graph. This fact is very important to our study because we are interested in the features that the system craft to perfom the given task. We want to teach the system to excel at our task of interest and then use its knowledge to improve another task.

Another nice feature that TensorFlow brings into the table is its ability to prune the execution graph before starting its computations. Usually several sets operations are repeated along the graph, by detecting this, the system can automatically replace all incidences of each repeated sub-graph with a single one thus, saving memory and time. The same case happens with communication nodes. If several nodes from a single device are consuming the same data from another device, the system is prepared to detect this and ensure that the data transfer occur only once.

The framework code is deeply optimized. Implementations of the same interface target the different dispositives that the code can run in. It is built upon known mature frameworks such as cuDNN, a library for deep neural networks that targets NVIDIA GPUs, and Eigen, a C++ library for linear algebra, which was extended to offer tensor arithmetic support. On top of this infrastructure, TensorFlow offers a Python client which is very convenient for fast development.

An interesting tool that is packed with the framework is TensorBoard. With the huge complexity that machine learning models offer at this scale, it is important to know what is happenining at any point of the training process. TensorBoard offers a glimpse of how does the architecture for a particualar computation graph looks like. This will become handy later when describing our model.

There exist several systems that offer similar features to TensorFlow. Theano, Torch, Caffe, Keras to name a few. The purpose of this work is by no means to study the advantages or disadvantages of these systems nor to create a benchmark on their perfomance. As the pipeline was already written in Python, we choose TensorFlow to minimize the gluing code. Among the examples that come with the system there is one end to end example of how to transfer learining from one trained net to another task.

Chapter 3

Architecture

> We may say, roughly, that a mathematical idea is 'significant' if it can be connectted, in a natural and illuminating way, with a large complex of other mathematical ideas.
>
> A Mathematician's Apology
> G. H. Hardy

In this chapter, we will talk about the implementation of our experiment. The details of the pipeline architecture, and the techniques used to obtain and curate data are unveiled. Details on the data munging and preprocessing are also given. The backend was implemented in Python, while the clients are implemented in javascript.

3.1 Ingest

We built a system to ingest the images from the NOAA service. It lazily downloads the images by checking first if the file is already present in the temporary folder. If the file does not exist it downloads it, then the system tries to add it to the database and persistent file system. To maintain a coherent one to one mapping between the database and the file system, the process of adding a new scene must be successful both in the database and in the filesystem, otherwise, the file is erased from both, and the state of the system remains as it was before the ingestion attempt.

3.2 Tag

Aerial tagged data is scarce. In particular, for the purpose of our experiment, we don't have any useful metadata on the images. We propose a method to tag samples of the scenes using crowd sourcing. We built a service that crops samples from the images and exposes them to an online application that lets any user with access to tag an image. We have three categories: the image has water in it, the image does not have water in it, and it is not possible to tell. When a positive answer is obtained, the system persists de image in the data base with the information of from which scene was it extracted.

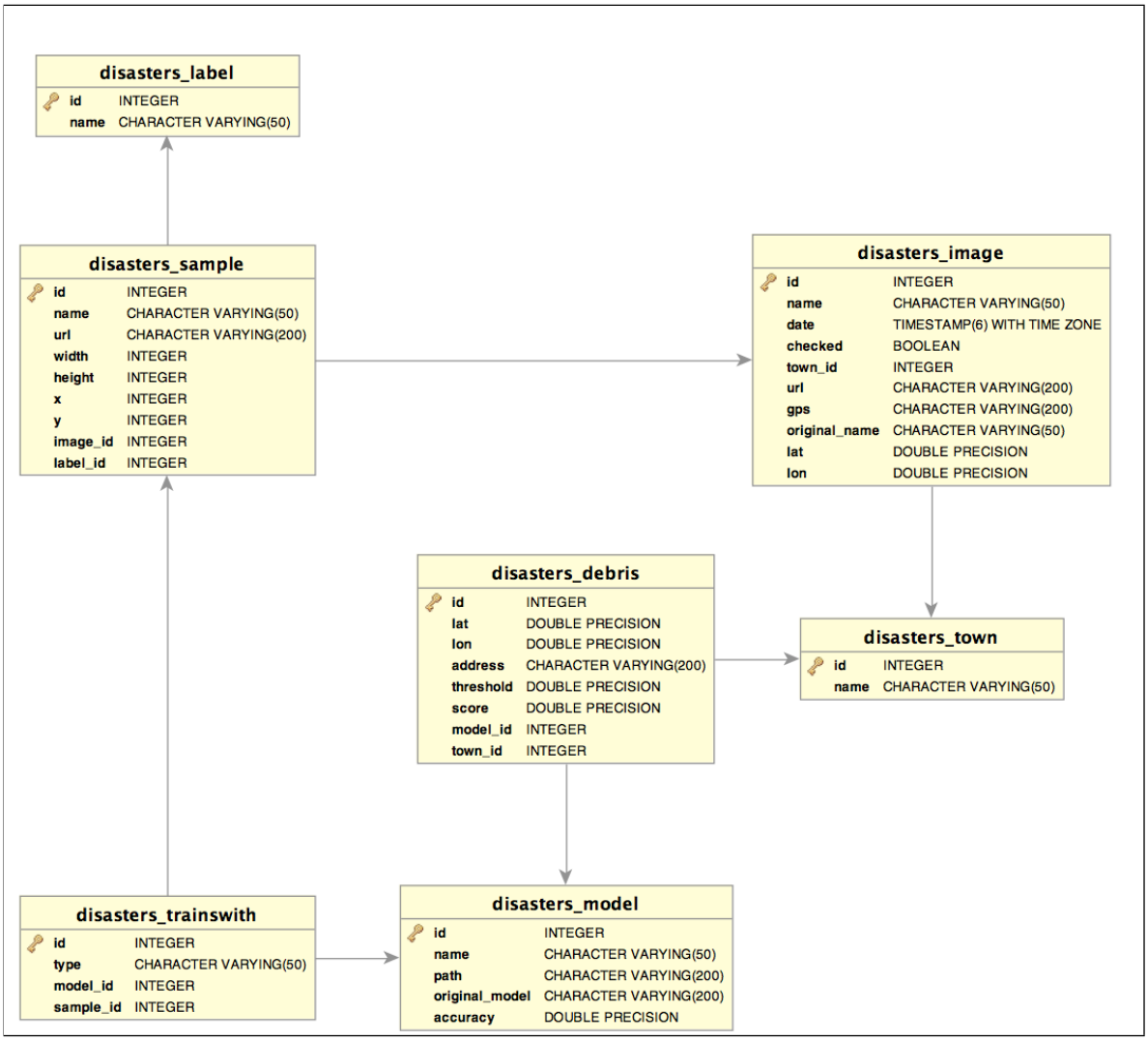
3.3 Data augmentation

Given the nature of our task, it is hard to acquire the tagged images. To increment the size of our training dataset, even more, we use a technique known as data augmentation. It relies on the fact that affine transformations do not change the content of the scenes, however, a transformed scene appears as a completely new one to the classifier.

The images were rotated by 10 degrees, and reflected by the x-axis and the y-axis this gives us a $x144$ factor, this means that for each tagged image, the training corpus is incremented by 144 images. The problem with this approach is that when a square image is rotated, some information on the corners is lost so we have to adjust the original image so that we can still crop a complete square from the desired size from it. For our experiment, the input size for the neural network is 227×227 pixels, so the original images must be at least $\sqrt{2}$ times 227 on each side. This way no matter how we rotate the original image, we can still crop a 227×227 from the center of the rotation without losing any data.

3.4 Database

In the figure we show a subset of the database diagram. Tables inherent to the features of the django framework are not shown. The database design was based on reproducible reaseach. We wanted to keep track of the charactistics of the models trained. Also we wanted to know which where the training images used for each model. In order for the tagging application to work, images must be ingested into the system so every time a new sample is tagged we populate the database with information about the original image and the coordinates relative to the original image. In the final stage of the process we produce a list of potential damaged buildings which are also inserted into the database with geografical information and human readable address. We think that this can be helpfull to allocate resources in the most efficient way.



3.5 Train

We needed data to train our model. Raw dron images where obtained from the National Center for Disaster Prevention (CENAPRED). Drons flew over three towns in the state of Oaxaca producing 3733 images during several days. Images contain gps information about the place where they where taken, however, it is not posible to produce a one to one mapping from the pictures to georeferenced points. With this limitation it was not possible to locate possible damaged buildings from the raw images. However, the model does not need any geografical information to be trained as it relies only in the pixel intensities.

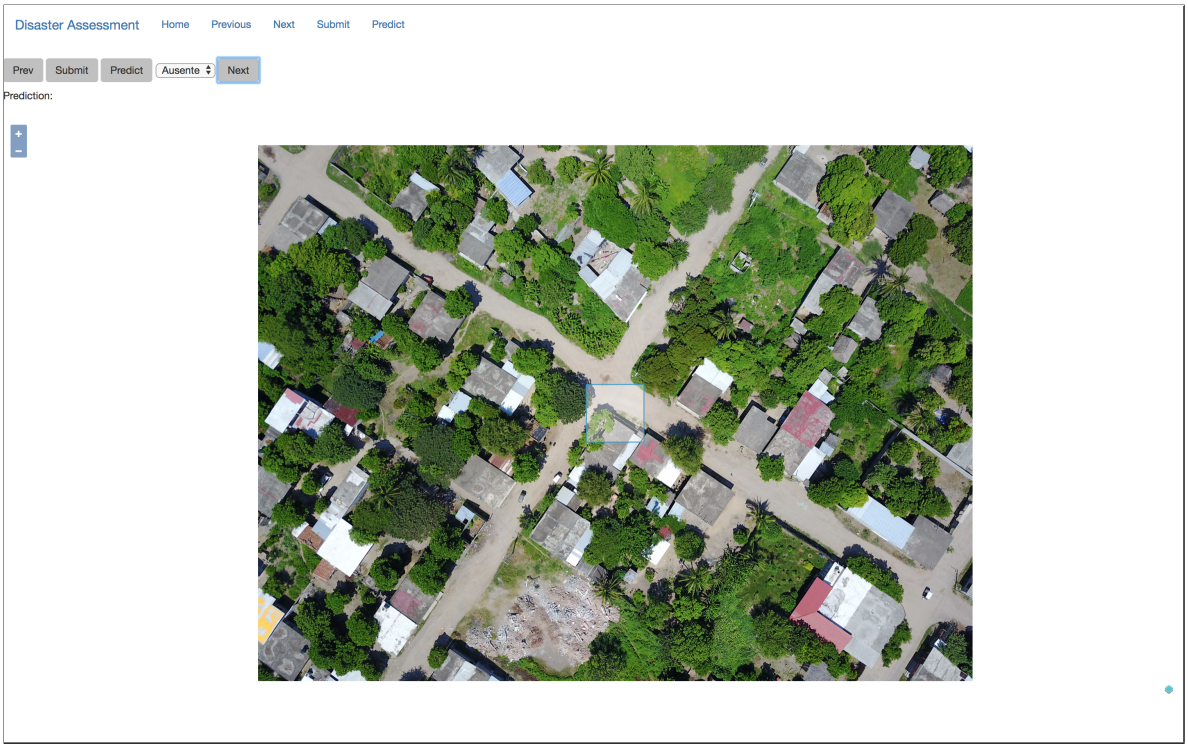
Cropping and tagging manualy a large number of samples from the images was prohibitive. In order to overcome this obstacle an online tagging application was developed for this purpose. The idea was to decentralize the tagging procedure by giving an easy to use tool that was able to run from any browser. This way the cumbersome task of tagging the images can be crowdsourced.

By consuming the REST api described in the previous section, a simple web client was developed using jquery and openlayers. The interface is an image viewer with a selection and a button to submit an opition on that the higlighted area. The user will select an appropiate section of the given image, tag it, and then submit the section. In the backend, the image is cropped and ingested into the database.

An additional button is given that lets the user see how well is the current best model performing. The process is quite similar to the previous one. In the server, the image is cropped and then the thumbnail is exposed to the model and the result is written back to the client through the REST api.

During the process, we noticed that the tagging process can lead to errors. In order to deal with those mistaken tagged images, a visualiser for the tagged images was developed. It consist in a simple interface where the last tagged images are shown with their respective tag. The interface offers a way to delete a given image or just edit the previously assigned tag.

Finally, we developed a view for the points that the algorithm finds in the ortorectified image. This view was implemented with open layers and shows information about the location of the potential damaged building in a human readable form. This is done by querying the google api with the latitude and longitude points extracted from the map.



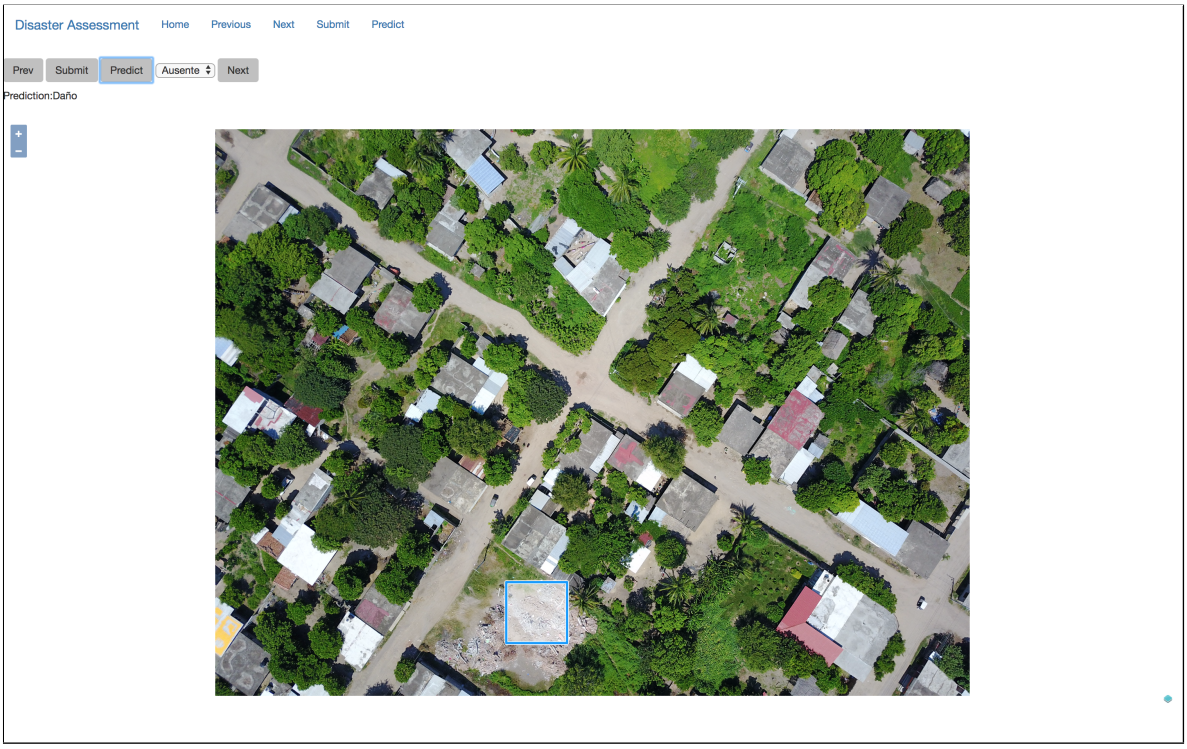
3.6 Predict

Another simple front end application was developed to predict on new features, it is very similar to the tagging application. Instead of asking the user about the correct tag, it queries the model and exposes the answer to the front end.

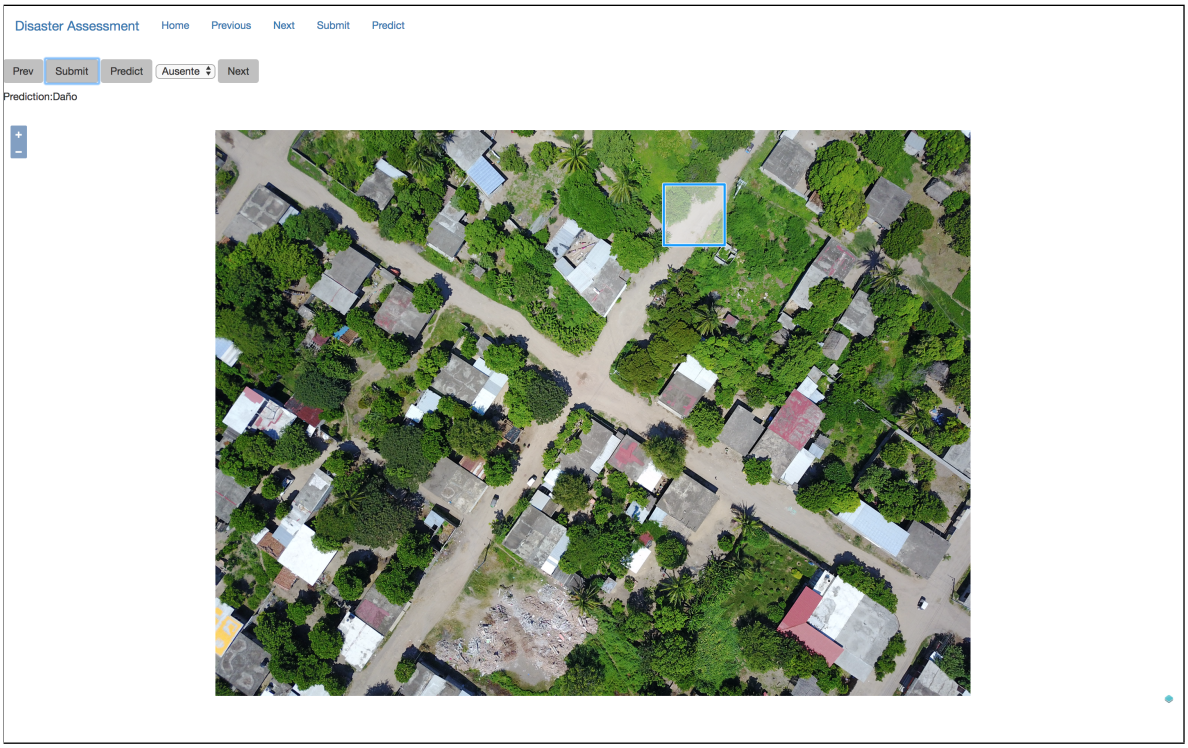
3.7 Model creation

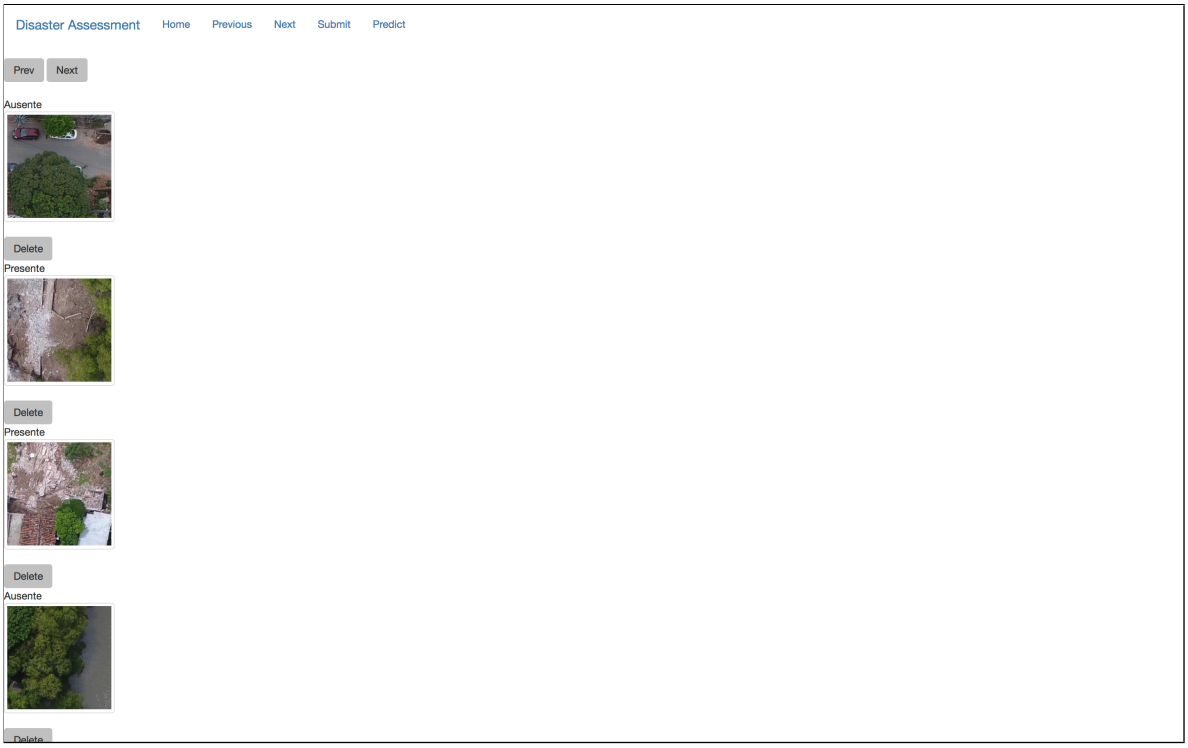
In order to create the model we need to select a sample from the thumbnails that we extracted from the original images. We wanted to create a process that was easily reproductible so we would compare models in a simple fashion. To this end the sample must be random each run and we need to split the images in three sets: training, validation and testing.

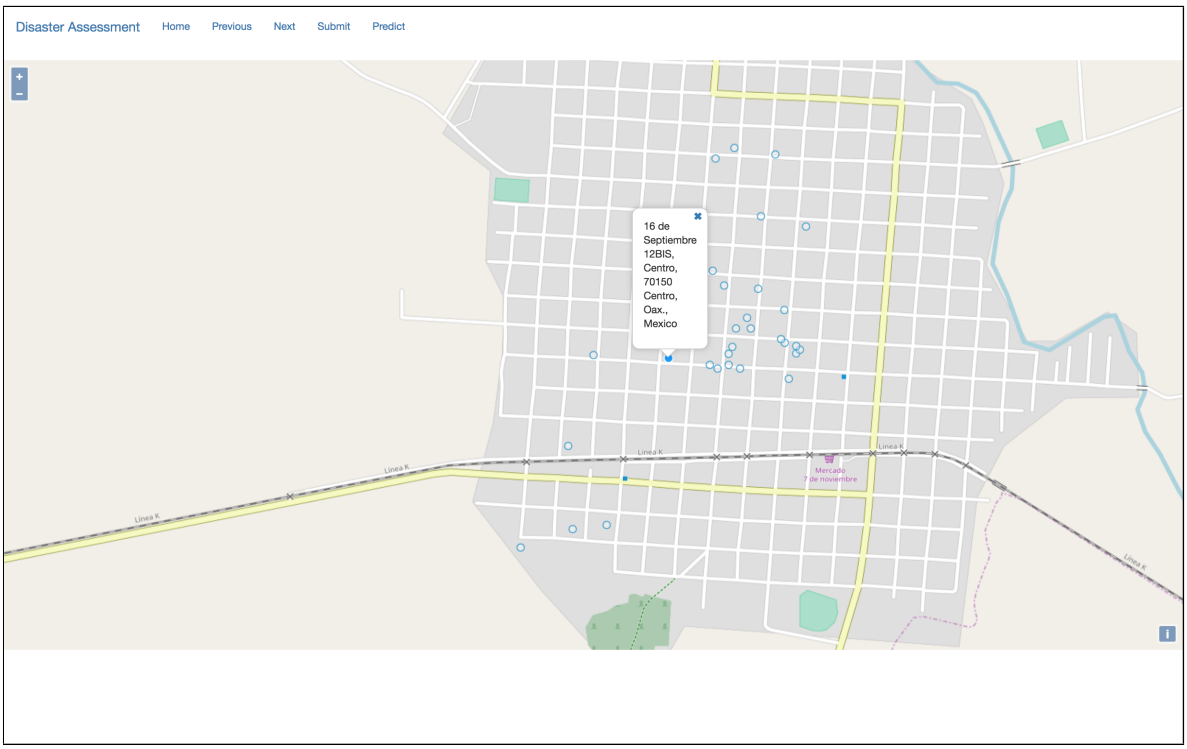
Tensorflow provides a script to retrain the last layer of inception by conecting the extracted features into a sofmax layer, and then training this classifier on the given set. It requires a directory layout tailored built to this purpose. The script was modified



to fit our database design in order to make as easy as possible to train several models with homogeneous training, validation, and testing sets.


Disaster Assessment
Home
Previous
Next
Submit
Predict
Prev
Submit
Predict
Ausente
Next
Prediction:Daño


Disaster Assessment
Home
Previous
Next
Submit
Predict
Prev
Next
Ausente
Delete
Presente
Delete
Presente
Delete
Ausente
Delete


Disaster Assessment
Home
Previous
Next
Submit
Predict
16 de Septiembre 12BIS, Centro, 70150 Centro, Oax., Mexico
Linea K
Linea K
Linea K
Linea K
Linea K
Mercado 7 de noviembre

Chapter 4

Analysis

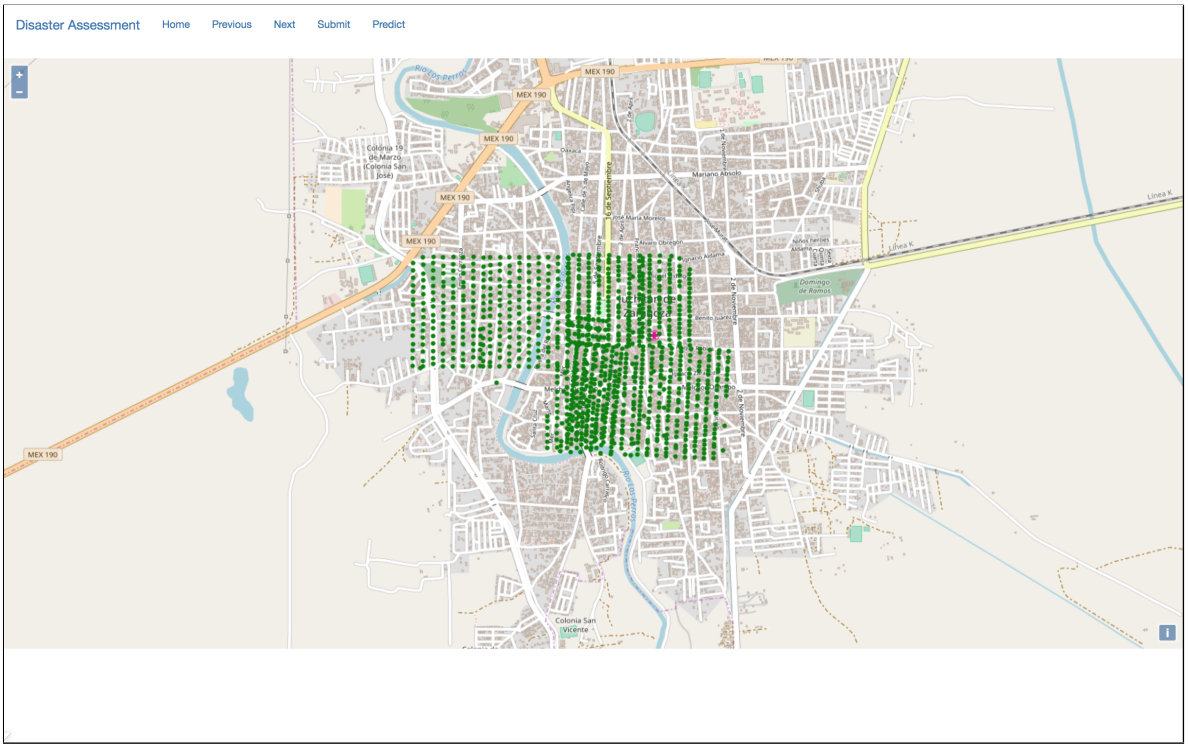
> A mathematical proof should resemble a simple and clear-cut constellation, not a scattered cluster in the Milky Way.
>
> A Mathematician's Apology
> G. H. Hardy

This chapter elaborates on how the model was trained and compared with other techniques from classical computer vision.

4.1 Exploratory analysis

Drone images from three different towns in the state of Oaxaca where obtained from the National Center for Disaster Prevention (CENAPRED). This pictures where taken during the week following an earthquake that originated in the Pacific coast of the state. We have 727 images from Santa Maria Xadani, 1872 from Union Hidalgo, and 1134 images from Juchitan of Zaragoza. As we can see in the following figures, the drones fly in a regular pattern forming a lattice of points where the images are taken. It is natural to think that given the spatial distribution, that the images are spatialy clustered. We wanted to show that this clustering translates to the space of the information that the images contain. In order to do this, a usual technique was applied to our set of images.

A t-sne analysis was performed with the images, and it is shown in the figure. To this end, the information from the pixels of each images was flattened into a vector

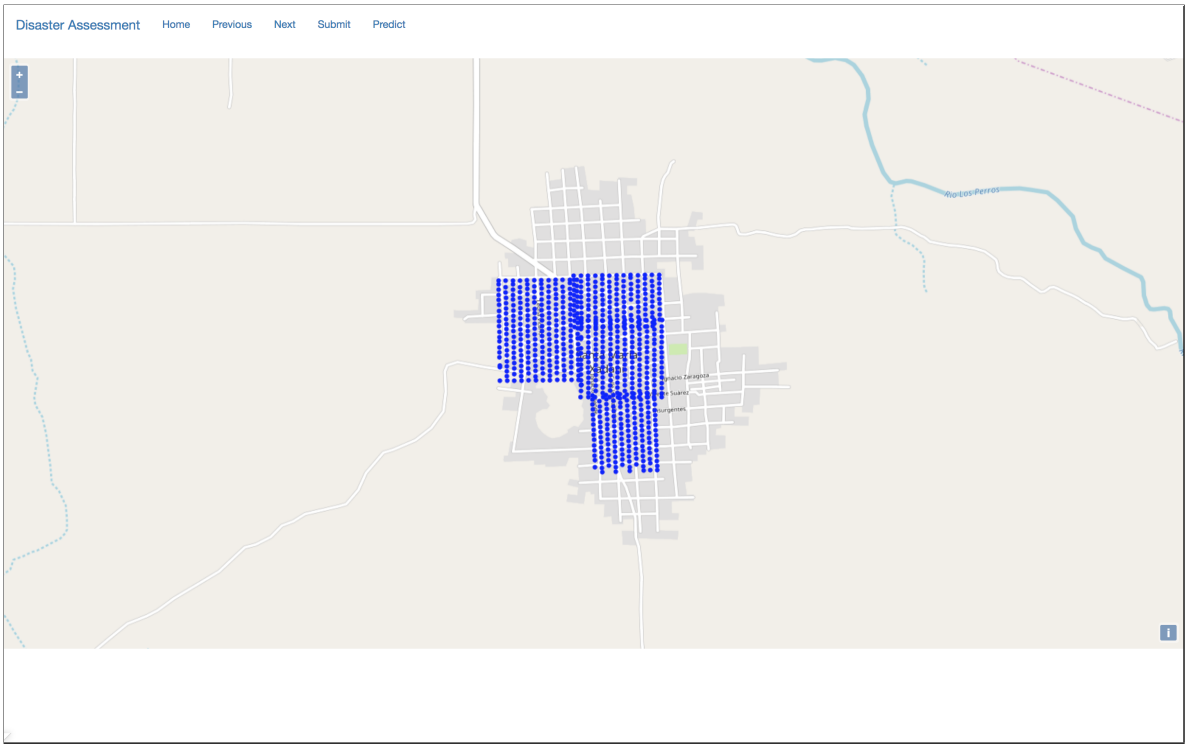


comprising the means and standard deviations. This was a simple dimensionality reduction technique and was useful to embed the images into a lower dimensional space. As we can see, images form natural clusters depending on the town that they where taken from. This was somewhat expected because of the light conditions during the exposure tend to have less variance among similar times and places.

The results obtained after applying t-sne, supports the proposed methodology of using images from one town, and try to predict on others. Due to the lack of training data, this was needed to be generated from the sample data. The application detailed in the previous chapter was used to crop and classify 100 square patches from the images in each of the towns. Each patch was 327×327 and a tag was assigined manualy by the author in each of them.

4.2 Model validation

In order to have an effective algorithm we need that it can perform well in places with images it has never seen before. To test this hipothesis we tried two different methodologies based on cross validation. First of all, classic n-fold cross validation was perform on the pool of training data generated for this experiment. This is, the training



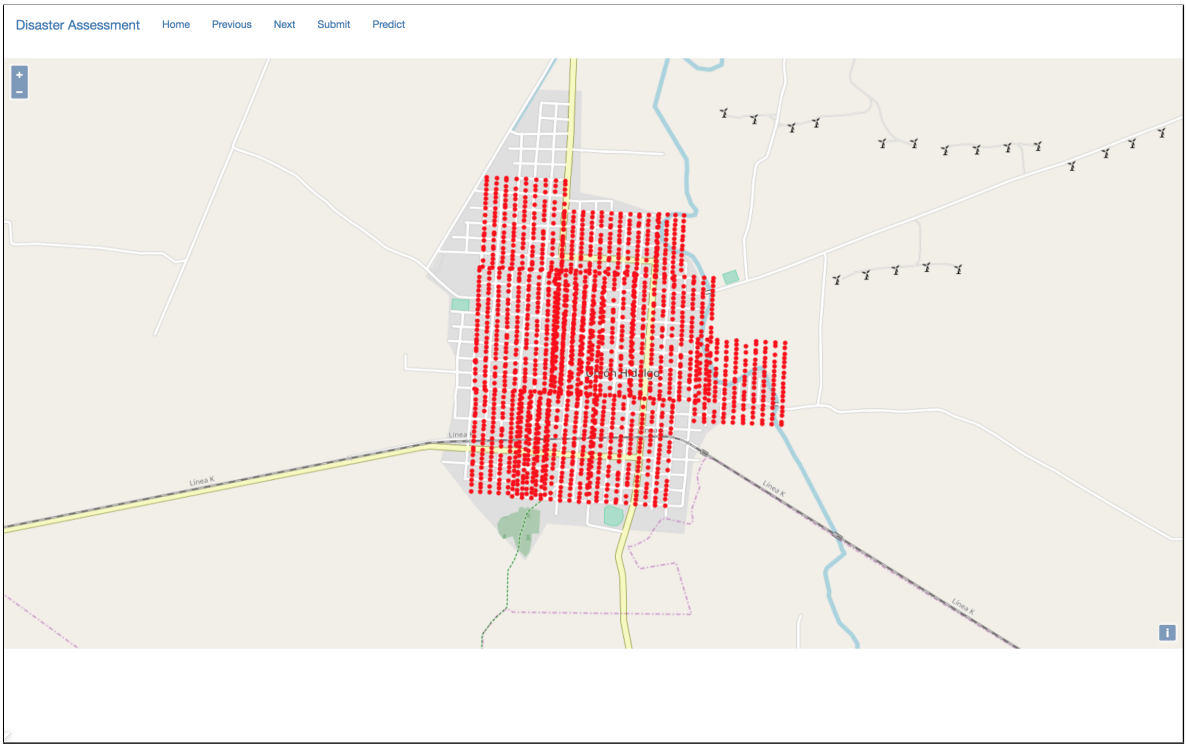
set was divided in n subsets and then a model was trained using $n-1$ of those sets while the remaining one was used for testing the model. This was repeated n times leaving a different set each time for testing.

4.3 Threshold selection

The binary classifier assigns a real number in the interval $[0, 1]$. To decide which values will be assigined with either level a threshold must be chosen. This was picked using a ROC curve. The ROC curve helps us chosing the performance that fits our needs in the ver possible way. For instance

4.4 How much is enough

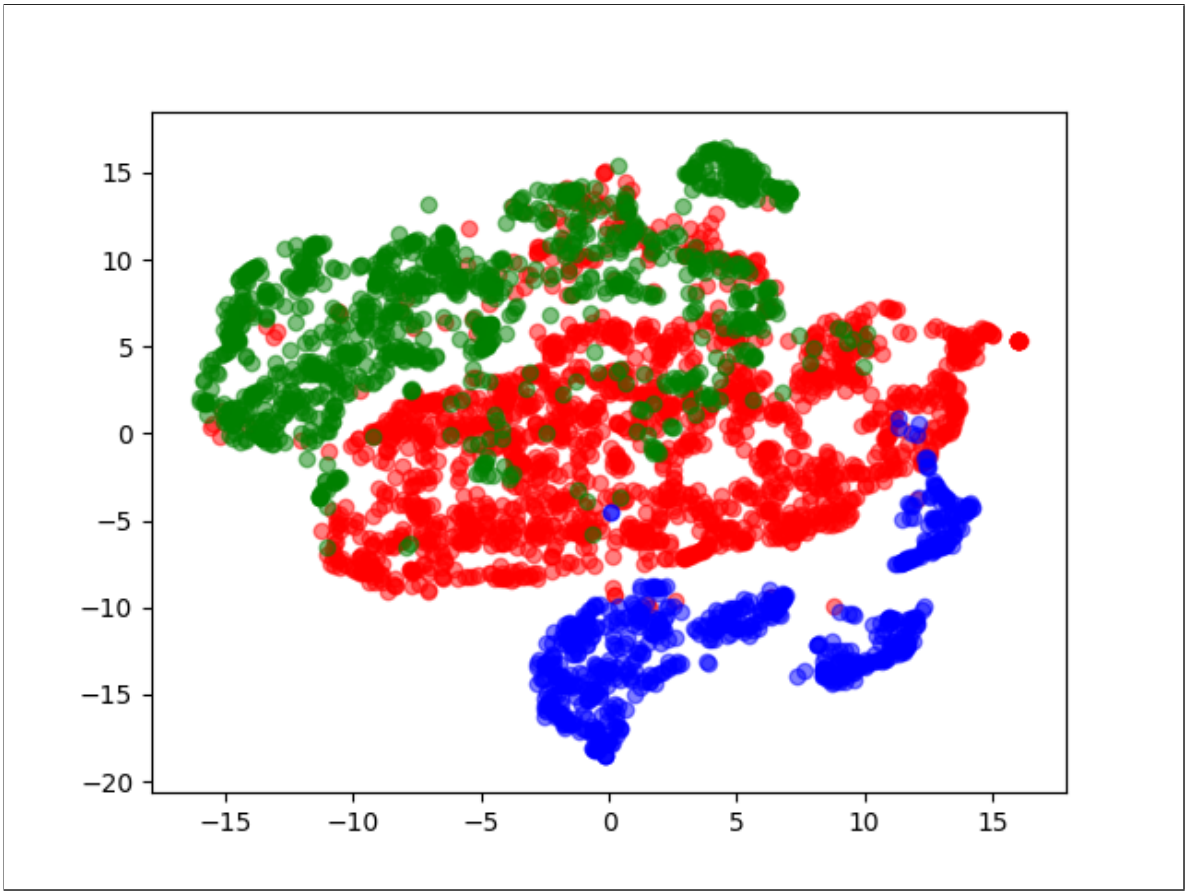
In this section we want to create a benchmark on how much images are needed to perform a retraining of the Inception network. What we wanted to achieve was to demonstrate that it is possible to obtain high accuracys using only a handful of images. We where able to test this by designing an experiment that measures the acurracy of



models trained with different sizes of training sets and testing them in a common test set.

4.5 Computer vision versus convolutional neural networks

4.6 Results


15
10
5
0
−5
−10
−15
−20
−15
−10
−5
0
5
10
15

Chapter 5

A glimpse into the future

> A mathematican, like a painter or a poet, is a maker of patterns. The mathematician's patterns, like painter's or the poet's, must be *beautiful*; the ideas, like the colours or the words, must fit together in a harmonious way.
>
> A Mathematician's Apology
> G. H. Hardy

This was an important experiment because it let us describe the use of novel techniques in the process of image classification.

5.1 Conclusion

Our efforts showed that it is possible to deliver a preliminary product with little training effort. This product might be used as a baseline in the event of a disaster such as an earthquake to give an idea of where to start looking for damaged buildings so resources can be allocated in an efficient manner.

5.2 Drawbacks

We noticed that, even when the classifier performs well in environments different to the one that it was trained with, it learns to classify dribble, not exactly damaged buildings.

We noticed some examples in which the classifier correctly finds scenes with presence of dribble, however, when we inspected the place using Google Street View, we noticed that there was no building in that place. This can can think of two possible scenarios in which this is possible; there was never a building in that place an it was used as a disposal for dribble from other places, or a house was built after the picture in Google Street View was taken. Bothe cases show inherent limitations of the methodology that we are proposing, we can only automate this kind of process to a certain extent.

5.3 Future work

An idea that was beyond the scope of this reasearch was to incorporate a technique known as active learning. In this scheme, the algorithm keeps improving as it receives feedback from the experts. In this fashion, when the model makes mistakes, this incorrectly labeled scenes can be relabeled and feed back to the system in order to create a new model and keep improving its performance. Although there is a limit to the extent in which the algorithm can perform, this might aid in some obvious cases that might not be present in the original training data.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. Tensorflow: A system for large-scale machine learning. *CoRR*, abs/1605.08695, 2016.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, abs/1511.00561, 2015.

[4] M.J. Canty. *Image Analysis, Classification and Change Detection in Remote Sensing: With Algorithms for ENVI/IDL and Python, Third Edition*. Taylor & Francis, 2014.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *CoRR*, abs/1412.7062, 2014.

[6] Y. Le Cun, O. Matan, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jacket, and H. S. Baird. Handwritten zip code recognition with multilayer networks. In *[1990] Proceedings. 10th International Conference on Pattern Recognition*, volume ii, pages 35–40 vol.2, Jun 1990.

[7] Jia Deng, Wei Dong, Richard Socher, Li jia Li, Kai Li, and Li Fei-fei. Imagenet: A large-scale hierarchical image database. In *In CVPR*, 2009.

[8] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *CoRR*, abs/1310.1531, 2013.

[9] Nashieli Garcia-Alaniz, Miguel Equihua, Octavio Prez-Maqueo, Julin Equihua Bentez, Pedro Maeda, Fernando Pardo Urrutia, Jos J Flores Martnez, Sergio A Villela Gaytn, and Michael Schmidt. The mexican national biodiversity and ecosystem degradation monitoring system. *Current Opinion in Environmental Sustainability*, 26:62 – 68, 2017.

[10] Steffen Gebhardt, Thilo Wehrmann, Miguel Angel Muoz Ruiz, Pedro Maeda, Jesse Bishop, Matthias Schramm, Rene Kopeinig, Oliver Cartus, Josef Kellndorfer, Rainer Ressl, Lucio Andrs Santos, and Michael Schmidt. Mad-mex: Automatic wall-to-wall land cover monitoring for the mexican redd-mrv program using all landsat data. *Remote Sensing*, 6(5):3923–3943, 2014.

[11] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[12] Lukasz Kaiser, Aidan N. Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *CoRR*, abs/1706.05137, 2017.

[13] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *CoRR*, abs/1511.02680, 2015.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou,

and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[15] Yury Kryvasheyeu, Haohui Chen, Nick Obradovich, Esteban Moro, Pascal Van Hentenryck, James Fowler, and Manuel Cebrian. Rapid assessment of disaster damage using social media activity. *Science Advances*, 2(3), 2016.

[16] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 14(5):778–782, May 2017.

[17] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014.

[18] Volodymyr Mnih and Geoffrey E. Hinton. Learning to detect roads in high-resolution aerial images. In *Proceedings of the 11th European Conference on Computer Vision: Part VI*, ECCV'10, pages 210–223, Berlin, Heidelberg, 2010. Springer-Verlag.

[19] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.

[20] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014.

[21] John A. Richards. *Remote Sensing Digital Image Analysis, Fifth Edition*. Springer-Verlag Berlin Heidelberg, 2013.

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[23] G. J. Scott, M. R. England, W. A. Starms, R. A. Marcum, and C. H. Davis. Training deep convolutional neural networks for land-cover classification of high-resolution imagery. *IEEE Geoscience and Remote Sensing Letters*, 14(4):549–553, April 2017.

[24] Brandt Tso and Paul M. Mather. *Classification methods for remotely sensed data*. CRC Press, Boca Raton, 2009.

[25] Michael Xie, Neal Jean, Marshall Burke, David Lobell, and Stefano Ermon. Transfer learning from deep features for remote sensing and poverty mapping. *CoRR*, abs/1510.00098, 2015.

[26] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014.

[27] X. Zhou and S. Prasad. Active and semisupervised learning with morphological component analysis for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 14(8):1348–1352, Aug 2017.