FORMAL SEMANTICS PROJECT REPORT

Jin Huang, Benjamin Beilharz – {huang, beilharz}@cl.uni-heidelberg.de

---

"Emotion detection on Social Media"

I. Motivation:

We are generally interested in tasks as sentiment classification, however, we wanted to go even deeper than binary classification as negative and positive. As we did not discuss emotion detection in class, yet heard about the colloquium of Saif Mohammed and we had an idea to start brainstorming. As this project was conducted while being in the midst of the onset of the COVID-19 virus, we got the topic really quickly. Given the recent events with COVID-19, we want to get a better understanding of the situation on social media (Twitter) and how people feel about their lives given the circumstances. This should be done by visualizing the overall emotion based on the VAD (Valence-Arousal-Dominance) scale which is a continuous three-dimensional space to measure overall sentence-inducing emotion.

II. Related Work

Most of the inspiration of this experiment was taken, from applying learnings of the semantic class, plus personal interests. We used methods as pre-trained word embeddings, neural network architectures and lexica we already knew about, to try a vague approach, but were able to learn and grow a lot.

III. Implementation Outline

Emotion Lexica

Emotion lexica were used as annotations for our tweets. We used the following of Saif Mohammad:

- VAD: http://saifmohammad.com/WebPages/nrc-vad.html
- EmoLex: http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm

VAD is a three-dimensional emotion space, such that emotions can be categorized as continuous values between three axes:

- valence – pleasure, displeasure
- arousal – excited, calm
- dominance – powerful, weak

The annotations were human crafted, such that we suspect them to be as individual as we see them, but the overall annotator agreement seemed to be promising enough. Ekman's basic emotions were captured within that space.
https://www.researchgate.net/publication/316546415_EmoBank_Studying_the_Impact_of_Annotation_Perspective_and_Representation_Format_on_Dimensional_Emotion_Analysis

EmoLex is a lexicon with emotion and sentiment (positive; negative) annotation on word-level which we further used in our features.

Data Retrieval

We decided to build our dataset from scratch, hence we wanted to work with recent data which was done by collecting data using the streaming API from Twitter, tweepy, where we acquired tweets and user data as JSON files. This said, we encountered the first problem; we needed account credentials for the API provided by Twitter, which seemed to be more of an obstacle for our second account as we thought. The application for the second account was refused such that we only had the option to stream data with one account. For this step, we wrote custom JSON configurations which we parsed for the setup to be flexible in which attribute-value pairs we want to work with and more importantly the content we were looking for in the tweets itself. Our choice became quite obvious picking topics we are also dealing with, like working from home, home-office, social distancing, loneliness, etc. Lastly, we decided that we take the tweet text, as well as other user-based information such as the location from the tweet JSON object. We serialized the retrieved JSON object to separate files which we later loaded into pandas data frames to ensure ease of processing for further steps.

Dataset Generation

After loading the JSON files into data frames we started to view what problems we might encounter when processing the data and a few things were obvious, to begin with:

- emojis
- user mentions
- hashtags
- retweets (they often did not include additional text)

After removing the mentioned with regex and encoding magic, we additionally removed digits as they do not hold any emotional connotation. Whitespaces, urls, stopwords and tweets that were not labeled as english were also dropped from the dataframe. This left us with a somewhat *good* corpus to start with.

As we did not know which approach we wanted to take, we also extracted some linguistic features as part-of-speech tags and entities using spaCy and it's large web corpus.

Corpus and Data Description

Our final corpus consisted of **12181 tweets**, with an **average length of 11 tokens**, whereas the vocabulary measured **13286 unique tokens**.
Furthermore we conducted statistical analysis on the coverage of our vocabulary with the annotation in the lexica.

For the **VAD lexicon** we found that:

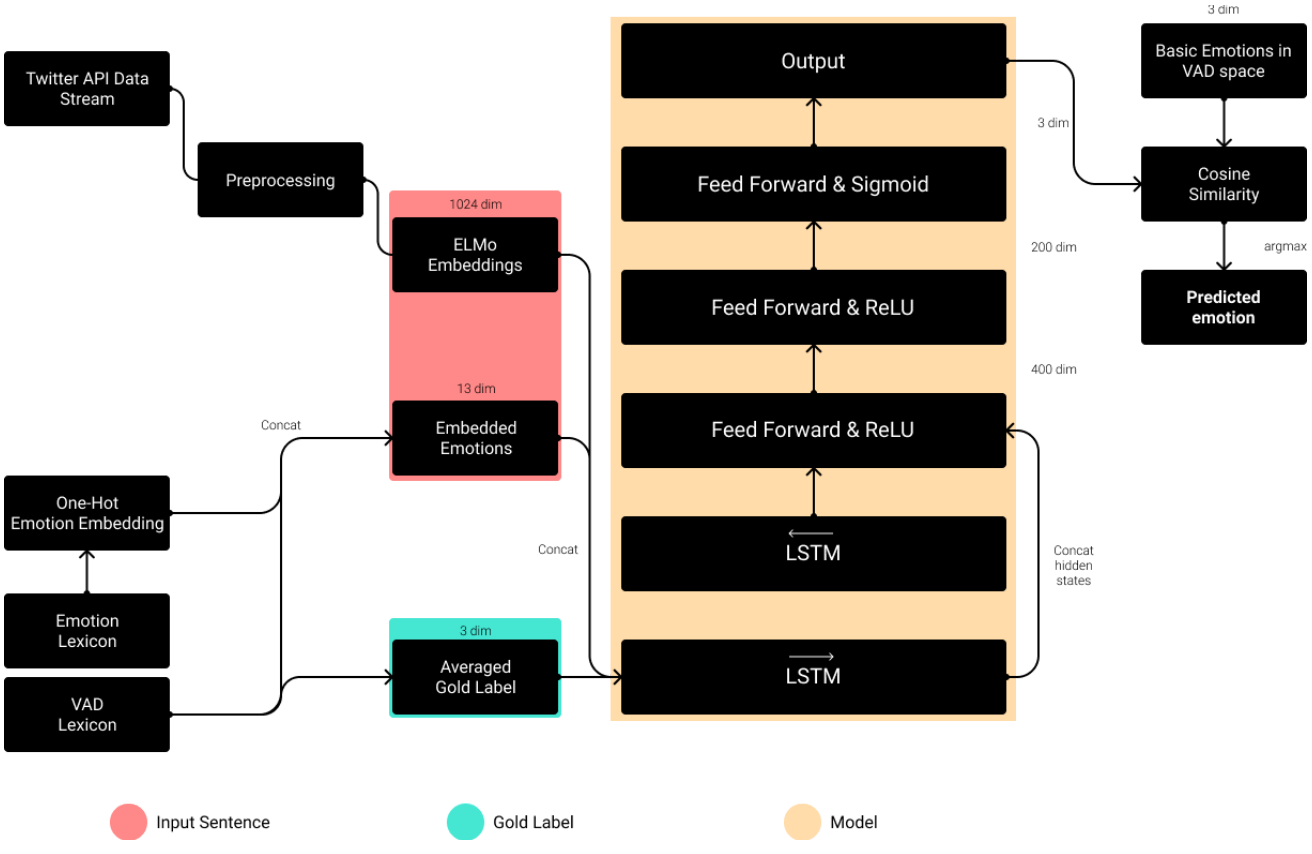
- **~27,3%** of our tokens had a lexicon entry
- **~91,8%** of our tweets had at least one token with a **VAD annotation**

Additionally the **emotion lexicon** had:

- **~30,64%** of the token coverage
- **~86,20%** of our tweets with at least one token with an **emotion annotation**

Network Architecture and Task Pipeline

We built our model using PyTorch. We took the knowledge of autoencoding (auto-encoder networks) into account and adjusted the method to our experiment, such that we use our LSTM to reduce and compress information of our high dimensional inputs to a three-dimensional vector for further classification. For our output layer we used sigmoid to squeeze the values into the range of 0 and 1 (just as in the VAD scaling).



Lexical Embeddings

Annotations were automatically extracted and serialized into a dictionary lookup for later purposes, whereas the words and the annotations are key and value respectively. For the VAD lexicon, we extracted the values as they were, as a three-dimensional vector describing their position in the VAD space. The emotion lexicon had 10 annotations for each word, consisting of emotions as anger, anticipation, disgust, fear, joy, sadness, surprise, trust plus annotations of positivity and negativity which were all extracted as one-hot encodings.

The average of our lexical embeddings was determined by taking the mean of the vector over the complete vocabulary.

EmoLex:

anger:	0.0207032
anticipation:	0.0346107
disgust:	0.01189813
fear:	0.03166815
joy:	0.02439832
sadness:	0.04587667
surprise:	0.0618462
trust:	0.02132783
positivity:	0.01532233
negativity:	0.04036033

VAD:

valence:	0.2900091
arousal:	0.20967007
dominance:	0.25763879

ELMo Embeddings

Instead of training embeddings from scratch, we decided to create sentence embeddings using the **ELMo-Embedder** by *AllenNLP*, which we fed our tweets into. We only used the last layer of the ELMo embeddings as previous works show that semantic information is mainly kept there.

Our final inputs *(x)* consist of three separate embeddings which are all concatenated, namely: the **ELMo embedding (1024 dims), categorical emotion lexicon embedding (10 dims) and the continuous VAD embedding (3 dims).** The latter was used to create the *pseudo ground truth* to calculate a loss for the backpropagation signal. We only lemmatized tokens if there was no lexicon entry available (to increase annotation coverage), elsewise the

word would have just gotten a zero-embedding. A sample input vector for a tweet is therefore a tensor of the dimensionality **batch-size x sequence-length x 1037**.

Pseudo Ground Truth

$$y_{tweet} = \frac{1}{|tweet|} \cdot \sum_{i}^{|tweet|} VAD(w_i)$$

For our autoencoding we had the task of predicting the overall sense of emotion, or the emotion of the tweet itself. The artificial/pseudo ground truth was computed by averaging the VAD vector over the words in the tweet, resulting in a three dimensional vector and can be compared to the output of our model. The output of our model was compared to the averaged VAD vector with which we could calculate the loss.

For classification we took Ekman's basic emotions and picked their values from the VAD dictionary (which is the prediction target space).

Basic Emotions (VAD Scale):

	Valence	Arousal	Dominance
Fear	**0.073**	**0.84**	**0.293**
Anger	**0.167**	**0.865**	**0.657**
Disgust	**0.052**	**0.775**	**0.317**
Joy	**0.98**	**0.824**	**0.794**
Surprise	**0.875**	**0.875**	**0.164**
Sadness	**0.052**	**0.288**	**0.164**

The final classification of the emotion was made via computation of cosine similarity of the models output to the above base emotions, taking the argmax to get the classified emotion.

Label Distribution

The following label distribution has been determined by a randomized train-dev-test split.

	Fear	Anger	Disgust	Joy	Surprise	Sadness
Train (80%)	7884	1217	344	283	12	-
Dev (10%)	1016	130	38	33	1	-
Test (10%)	1017	132	32	37	-	-

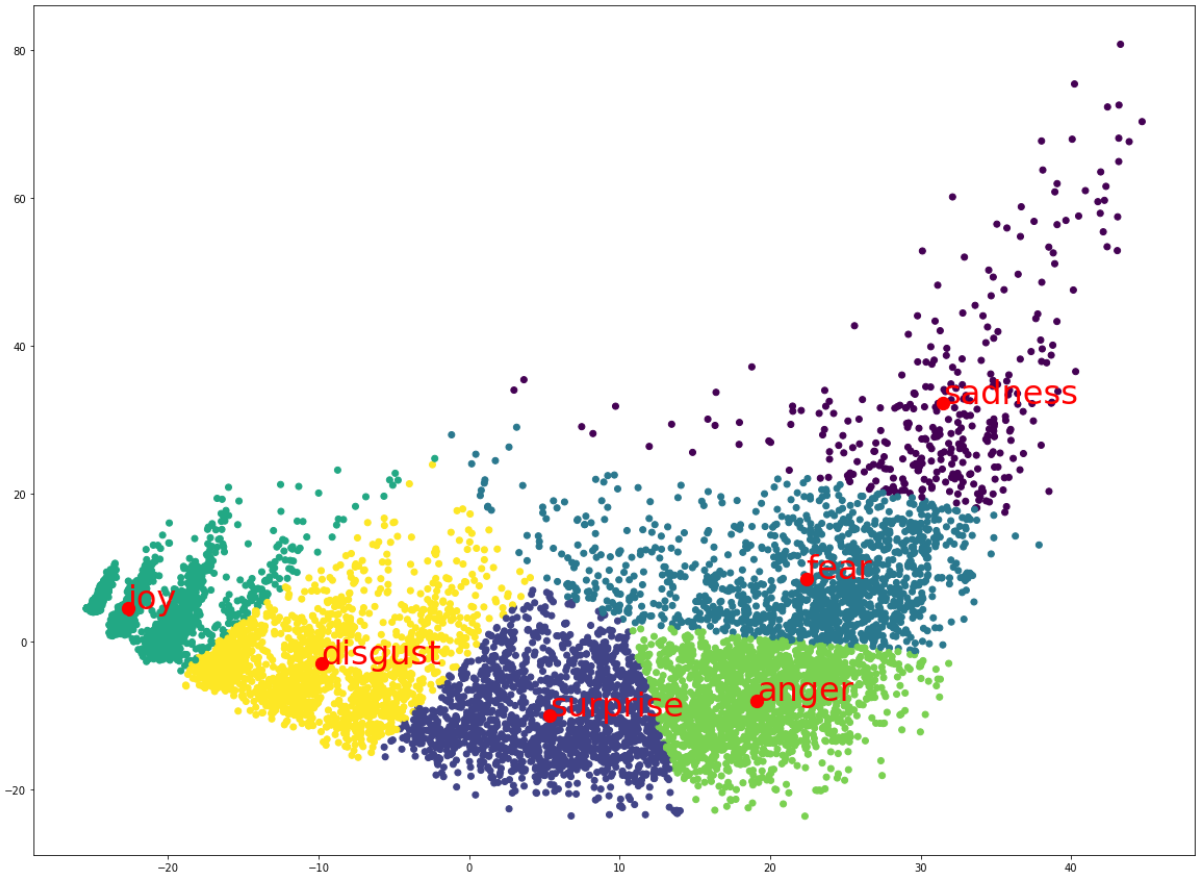
Baseline

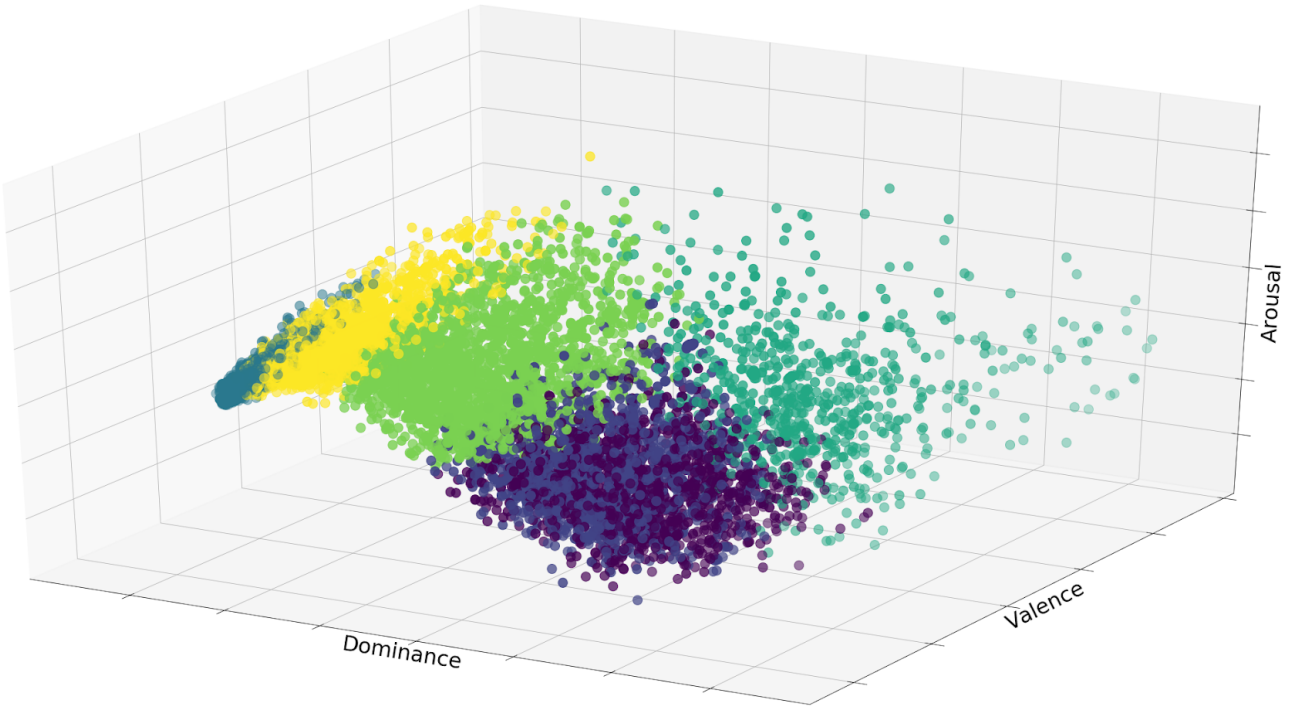
As our approach shows that we use a neural network to densify the information of the previous input vector (the goal was to compress the information of the semantic layer and the emotion encoding into a three-dimensional vector), which should be able to be used for classification over the least distance. We might think of our base dimensions and their vectors as some kind of artificial and fixed centroids, whereas our computed and compressed vectors are the points in the three-dimensional space that seek belonging through least distance to those centroids. We classified with unsupervised learning via KMeans, assuming six individual clusters (for each of the base-emotions).

Furthermore, after thinking of a baseline, we could also have taken the labels generated through clustering as our supervision signal, not using an autoencoder, but as a simple classification task at hand, applying softmax to find the probability for each sample.

Labels via clustering & cosine similarity

	Fear	Anger	Disgust	Joy	Surprise	Sadness
Count	2868	1614	695	1683	969	1911


sadness
fear
joy
disgust
surprise
anger


Dominance
Valence
Arousal

Training

Hyperparameter Settings

Parameter	Embedding Dimension	Batch Size	Hidden Size	#Layers	Epochs	Optimizer	Learning Rate
Value	1037	32	100	2 (bidirectional)	5	ADAM	0.001

III. Results

The results were not quite as expected given the loss. After testing the model and letting it classify some excerpts from our corpus we have come to realize that our model didn't learn what we wanted it to learn. We computed the cosine distance for each basic emotion with the model vector output and classified the emotion by taking the maximum similarity. Unfortunately our results were lacklustre. We further applied the same method to the pseudo-ground truth we created with averaging over the input sentence. We can see that our model from experiment A only went for the *fear* classification, whereas our pseudo-labels were categorized in different emotions. Although our pseudo-labels were categorized differently, we see a huge bias towards the *fear* emotion, which brings us to the conclusion that we overfitted the model onto the training data even with applying some regularization techniques.

	Train	Dev	Test
Average Loss Std Loss	1.570 1.420	0.0033 0.0014	0.0036 0.0012
Accuracy	83.48%	81.85%	82.51%

Can we trust the result?
We annotated each 150 samples by hand to check our classification to the hand annotated samples. We test our gold annotation versus our prediction of our model and versus our heuristic.

Confusion Matrices

The F1 score of the pseudo-gold labels and predicted labels seem satisfying, mostly because the classification is very unbalanced, almost 78% sentences were classified as “fear”, rarely as “anger”, “disgust”, “joy”, and none as “surprise”, “sadness” and “neutral”. However we also did not have the class “neutral” during Pseudo-Gold labeling and training, and that’s why none of tweets were classified as “neutral” here.
But if we compare the pseudo-gold labels to the (of two people) annotated labels we can clearly see that the pseudo-gold classification has already a very big difference to the annotations, same as the predicted classification, while the difference between the pseudo-gold labels and predicted labels is really small.
Comparing the two annotations we can see that the most annotated label is “neutral”, which was either used in the Gold-Pseudo or the predicted classification. Besides we can see that only 68% of the annotations are consistent. For a lot of natural language sentences we already have different interpretations as humans, to get a better understanding we need more annotators and improve our pseudo-gold classification.

Gold vs Prediction

	fear	anger	disgust	joy	surprise	sadness	neutral
fear	117	0	0	0	0	0	0
anger	16	0	0	9	0	0	0
disgust	5	0	1	0	0	0	0
joy	0	0	0	2	0	0	0
surprise	0	0	0	0	0	0	0
sadness	0	0	0	0	0	0	0
neutral	0	0	0	0	0	0	0

	TP	TN	FP	FN	F1
fear	117	12	0	21	0.917647
anger	0	125	25	0	0.000000
disgust	1	144	5	0	0.285714
joy	2	139	0	9	0.307692
surprise	0	150	0	0	NaN
sadness	0	150	0	0	NaN
neutral	0	150	0	0	NaN

Gold vs Annotator 1

	fear	anger	disgust	joy	surprise	sadness	neutral
fear	22	12	5	21	0	9	48
anger	4	2	0	1	0	1	17
disgust	0	1	0	1	0	1	3
joy	0	1	0	0	0	1	0
surprise	0	0	0	0	0	0	0
sadness	0	0	0	0	0	0	0
neutral	0	0	0	0	0	0	0

	TP	TN	FP	FN	F1
fear	22	29	95	4	0.307692
anger	2	111	23	14	0.097561
disgust	0	139	6	5	0.000000
joy	0	125	2	23	0.000000
surprise	0	150	0	0	NaN
sadness	0	138	0	12	0.000000
neutral	0	82	0	68	0.000000

Gold vs Annotator 2

	fear	anger	disgust	joy	surprise	sadness	neutral
fear	20	22	6	20	3	9	37
anger	3	3	1	0	1	0	17
disgust	1	0	0	0	1	1	3
joy	0	1	0	0	0	1	0
surprise	0	0	0	0	0	0	0
sadness	0	0	0	0	0	0	0
neutral	0	0	0	0	0	0	0

	TP	TN	FP	FN	F1
fear	20	29	97	4	0.283688
anger	3	102	22	23	0.117647
disgust	0	137	6	7	0.000000
joy	0	128	2	20	0.000000
surprise	0	145	0	5	0.000000
sadness	0	139	0	11	0.000000
neutral	0	93	0	57	0.000000

Prediction vs Annotator 1

	fear	anger	disgust	joy	surprise	sadness	neutral
fear	22	15	5	23	0	11	62
anger	0	0	0	0	0	0	0
disgust	0	0	0	0	0	0	1
joy	4	1	0	0	0	1	5
surprise	0	0	0	0	0	0	0
sadness	0	0	0	0	0	0	0
neutral	0	0	0	0	0	0	0

	TP	TN	FP	FN	F1
fear	22	8	116	4	0.268293
anger	0	134	0	16	0.000000
disgust	0	144	1	5	0.000000
joy	0	116	11	23	0.000000
surprise	0	150	0	0	NaN
sadness	0	138	0	12	0.000000
neutral	0	82	0	68	0.000000

Prediction vs Annotator 2

	fear	anger	disgust	joy	surprise	sadness	neutral
fear	21	25	7	20	5	10	50
anger	0	0	0	0	0	0	0
disgust	0	0	0	0	0	0	1
joy	3	1	0	0	0	1	6
surprise	0	0	0	0	0	0	0
sadness	0	0	0	0	0	0	0
neutral	0	0	0	0	0	0	0

	TP	TN	FP	FN	F1
fear	21	9	117	3	0.259259
anger	0	124	0	26	0.000000
disgust	0	142	1	7	0.000000
joy	0	119	11	20	0.000000
surprise	0	145	0	5	0.000000
sadness	0	139	0	11	0.000000
neutral	0	93	0	57	0.000000

```
------------------------------------------------------------------

Annotator 1 vs Annotator 2
           fear  anger  disgust  joy  surprise  sadness  neutral
fear         17      3        0    2         3        0        1
anger         1     12        1    0         0        1        1
disgust       0      1        4    0         0        0        0
joy           0      3        0   14         1        1        4
surprise      0      0        0    0         0        0        0
sadness       0      2        0    0         1        7        2
neutral       6      5        2    4         0        2       49

           TP   TN  FP  FN        F1
fear       17  117   9   7  0.680000
anger      12  120   4  14  0.571429
disgust     4  142   1   3  0.666667
joy        14  121   9   6  0.651163
surprise    0  145   0   5  0.000000
sadness     7  134   5   4  0.608696
neutral    49   74  19   8  0.784000

------------------------------------------------------------------
```

IV. Conclusion

Although we were not able to train a proper classifier, we take this project as a personal milestone in gathering experience in this discipline and are more certain than ever that good data is a luxury and scarce. We also have been able to gather experience with a few tools in deep learning like Pytorch and were able to build, navigate and monitor training processes (with tensorboard). Overall trying out unconventional ideas were quite refreshing, even if we would not be able to fake ground truth like we did in this project.