Style Transfer in Text

Anthony Alvarez, Fei-Tzin Lee, Elsbeth Turcan

aea2161, fl2301, ect2150 (Group 1)

Problem Statement

- A core concept in linguistics is the idea that there is more than one way to say the same thing. A blog post, a journal article, and a political speech might all convey the same information, but we would not expect them to sound or look the same—they might include more or fewer contractions, personal pronouns, active or passive verbs, and so on. We call these variations in how something is said *sociolinguistic style*.
- In our project, we would like to take one document D_{1S_1}, written in some style S_1, and a target style S_2, and output a new document D_{1S_2}, containing the same information content as D_{1S_1} but written in the style S_2. For example, we might rewrite Beatles lyrics in the style of Shakespearean plays.
- An application which had the ability to transform text between writing styles while preserving its content would be invaluable for sharing knowledge across academic disciplines, educating students, and disseminating technical work of any kind to the public.

Related Work

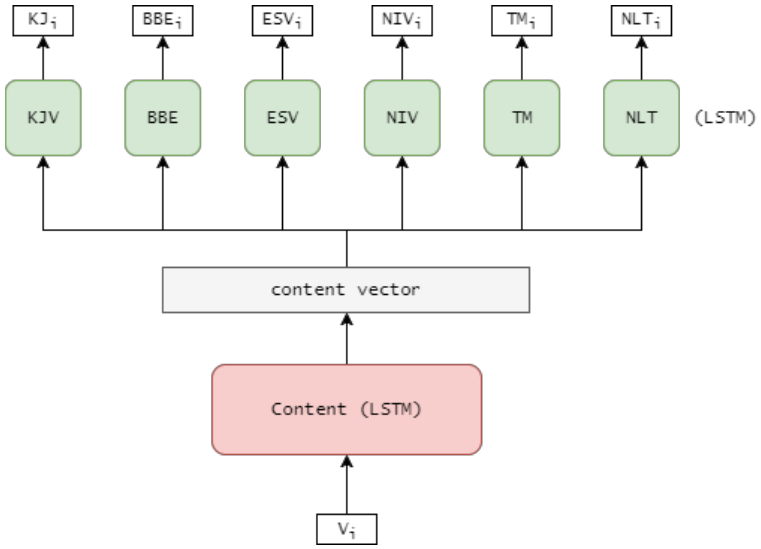
- Artistic style transfer has recently appeared in scientific literature as researchers have looked to convert one image into the artistic style of another (taking on qualities like color, line thickness, and so on) while preserving its content (the scene depicted in the first image) [2].
- This visually impressive phenomenon has been extended to domains like video [4]. The problem seems somewhat more difficult for text, however, because style information in images is much more abundant than style information in text.
- Kabbara and Cheung [3] proposed a variant of style transfer using recurrent networks instead of convolutional networks, although the work has not yet been published.
- We propose to accomplish style transfer using variational autoencoders following the impressive results of Bowman et al. [1] modeling sentences as soft regions in sentence space and traversing paths between them smoothly. We hope to use a similar model in order to smoothly traverse “style space”.

Preliminary Results

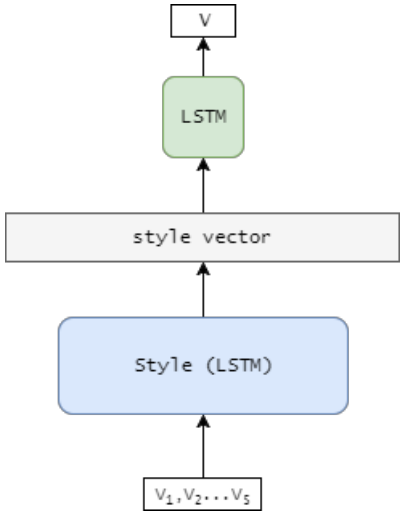
- We gathered a corpus of six parallel translations of the same text in order to differentiate style from content. We have collected six translations of the Bible (King James Version, Basic English, 2001 English Standard Version, 1984 New International Version, 2002 The Message, and 1996 New Living Translation), all broken into chapters and verses. We expect each translation to contain matching content but be written in a different style.
- We have devised an architecture and a pre-training scheme for this task, which are pictured on the next page. We will have two neural networks that we expect to encode "style" and "content", and we will pre-train them both separately with our Bible data. Then we will combine those networks to create a final architecture in which the "content" network takes in the document whose content we wish to preserve, the "style" network takes in some sample of the target style, and the output of the network will be the content of that document in the target style.

Architecture Diagrams – Pre-training

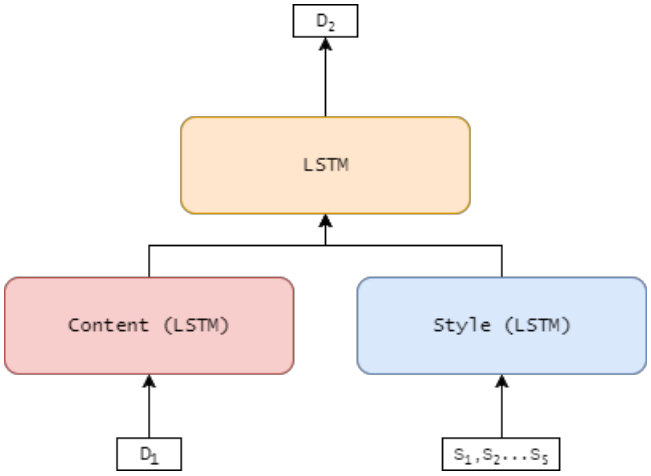
Content pre-training: take in some verse V_i and learn to output that same verse in all six "known" styles. Later discard the six "style" helper networks.



Style pre-training: take in a five-verse sample of some style V and learn to predict which of the six "known" styles it is. Later discard the helper classifier.



Architecture Diagrams - Final



Final design: Extract the pre-trained style and content networks and begin training.

Train the network with our dataset by feeding in combinations such as: one verse a_i to the content network, other verses $\{a_j, a_k, \ldots a_n\}$ from the same style to the style network, with the original verse a_i as the desired output; one verse a_i to the content network, five verses from a different style $\{b_j, b_k, \ldots b_n\}$ to the style network, and the corresponding verse in the target style b_i as the desired output, and so on.

Future Work

- Our data has been organized into an easily accessible JSON format but we intend to perform more data cleaning to eliminate things like unknown characters, as we have noticed these causing difficulty.
- We are presently at work implementing the architectures and will begin pre-training, followed by the actual training of the network, and then testing and refinement.
- Our present architecture will be trained on sentence-level or small chunk-level data. In future, it would be interesting to expand such a system to deal with entire documents and investigate whether the types of style and content representations that each system learns are comparable.

References

[1] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowics, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. *arXiv preprint arXiv:1511.06349*.

[2] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2016. Image Style Transfer Using Convolutional Neural Networks. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2414-2423.

[3] Jad Kabbara and Jackie Chi Kit Cheung. 2016. Stylistic Transfer in Natural Language Generation Systems Using Recurrent Neural Networks. In *Proceedings of EMNLP 2016 Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods,* 43-47. Association for Computational Linguistics.

[4] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. 2016. Artistic style transfer for videos. *arXiv preprint arXiv:1604.08610*.

Mixlingual ASR

Group 2
Emily Hua (yh2901), Kaili Chen (kc3031), Wanting Wang (ww2440)

Definition of the Problem

Code-switching is a phenomenon where there is more than one language within an utterance, which is common in many multilingual countries. The three team members are all proficient in Mandarin and English, yet fail to find a messaging app that provides mix-lingual speech to text functionalities. Hence, in this project, we aim to build an mix-lingual ASR using deep neural networks on a code-switch corpus.

Related Work

OC16-CE80: A Chinese-English Mixlingual Database and A Speech Recognition Baseline

The paper presented the data profile of OC16-CE80 Chinese-English mixlingual speech database, which was released to support the MixASR-CHEN challenge on O-COCOSDA 2016. This database consists of 80 hours of speech signals recorded from more than 1,400 speakers, where the utterances are in Chinese but each involves one or several English words. The speakers of OC16- CE80 are totally from the China mainland.

The paper also presented a speech recognition (ASR) baseline that was constructed with the deep neural network-hidden Markov model (DNN-HMM) hybrid system. The acoustic model (AM) was built largely following the Kaldi WSJ s5 nnet3 recipe. The language model (LM) was used the conventional 3-grams. Four LM configurations were investigated, where MIX LM works the best on both English and Chinese words. It leads to the lowest 19.00 WER on Chinese words, 43.67% WER on English, and 20.09% WER overall.

Preliminary Results

Data Source: LDC2015S04 SEAME (South East Asia Mandarin-English) corpus

It contains ~27 hours worth of conversational speech audios with transcription.

Toolkit: Kaldi (C++ and Shell script interface)

For data processing (done in Python and Shell script):

1. Acoustic Data Preparation: we have successfully extracted MFCC and statistics for cepstral mean and variance normalization.
2. Language Data Preparation: we filtered out ill-segmented sentences, created the customized lexicon based on CMU lexicon (EN) and THCHS30 lexicon (CN), and successfully generate the .fst file.
3. Successfully generated a monophone system
4. Decoding: we have gone through online decoding using kaldi example monophone models with THCHS30 datasets, a mandarin based dataset.

Future Plan

Now that we have prepared MFCC features, language models, training and test set, we're going to train and test different models: monophone, tri1, tri2, tri3, dnn, and etc. Most of the models will be conducted in kaldi. As for DNN model, we're going to conduct it on Keras, since it is much easier to add hidden layers and tune the parameters.

We will compare WER based on different models and optimize our algorithms. In the final step, we will perform online decoding on a recorded wav file as well as try to use PortAudio to decode our own voice if time permits.

Deep Learning Project

Richard Godden & Yogesh Garg
rg3047 & yg2482
Group 3

Project definition

- Problem 1:

  To convert music audio file to midi format

- Problem 2:

  To apply a style transfer on a midi file from one music genre to another

Related works

- The Lakh MIDI Dataset: dataset of midi files matched to million song database
- Music net: a curated collection of labeled classical music
- WaveNet: a Generative Model for Raw Audio
- An End-to-End Neural Network for Polyphonic Piano Music Transcription
- An Experimental Analysis of the Entanglement Problem in Neural-Network-based Music Transcription Systems: shows NNs learn combinations of notes, and have a hard time generalizing to unseen combinations of notes
- Further notes and methods and investigations can be found here

Preliminary Results

- We went through the Music Net code
  - We saw how the Midi Files are laid out, and how they can be used
  - We followed the instructions to train an MLP on it
  - We noticed the waveforms that the MLP learns, as suggested by the paper
- We also went throw the fastText code
  - We were able to train it on the sample texts that the authors provide
  - The word embeddings so generated made sense
- We checked out midicsv, a midi to csv converting library
  - It has a way of converting all midis to a list of events
  - We are trying to define a word out of these events
  - (See Future plan)

Future plan

- Problem 1:
  - We intend to learn an embedding for midi notes using the fastText skip-gram algorithm
  - We hope this will capture the semantic meaning of notes and their relationship to chords and scales
  - This is something that direct audio to midi algorithms appeared to have missed
  - We will then train a model to generate these embeddings from an audio file
- Problem 2 (Due to time constraints, this problem is out of scope for the project):
  - We can train different embeddings for different music styles and use those to classify or generate music pieces
  - use GANs trained on different styles conditional on the melody midi track then use the melody of one piece feed that into the GAN of a different style

Deep Learning For Flower Classification

Ignacio Aranguren, Rahul Rana, Xiaoxue Du
(ia2221, rr3087, xd2164)

Problem

The goal of the project is to classify different species of flowers.

- There are 369,000 species of flowering plants in the world.
- Flowers have a lot of inter-class similarity and intra-class variations, which makes classifying flowers a unique problem.
- People usually consult specialists or browse the web for learning the species.
- Better way to identify can be done by classifying flower images using mobiles.
- Flower classification has significant applications in computer vision and botanical research.

Related Work

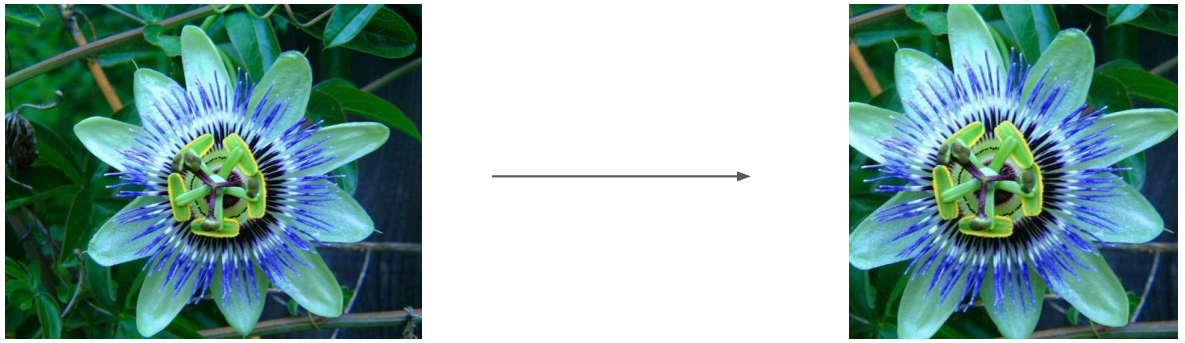
- There's an 18th-century hierarchical plant classification system that was proposed by Carl Linneaus and is still in use today.
- In traditional approaches to flower classification, botanists observe life habits of a flower and study its structure and morphological characteristics; much of this is experience-based and guided by domain experts.
- Nilsback and Zisserman [1] were amongst the first to apply machine learning techniques to the problem of flower classification; they described a flower in terms of its color, shape, and texture and then performed classification using an SVM classifier.
- Liu, Y., Tang, F., Zhou, D., Meng, Y., & Dong, W. [2] acheived the current state-of-the-art accuracy (November 2016) with flower species (84.02%), using a 7-layer convolutional network, and a custom designed input dataset, having 79 flower species. They have tested their model on the oxford dataset (102 species). They uniquely combined the luminance map and saliency map to the images to be increase inter class differences.

Preliminary Results

We are using the Oxford dataset (102 species); we're looking to compare our results with those of others who've worked on the same dataset previously.

An important piece to succeeding with this classification task is the pre-processing of the images so that we can later train a neural network and compare different architectures. We've been working on recognizing the area within an image that has the flower, and we've successfully done automatic segmentation of the images (boxing of flower). We've also automated the cropping and re-sizing of these segmented images to get the images into a form more amenable for training models on.

Below is an example of an image that has been automatically cropped by our pre-processing program.



We have set up a neural network closely inspired by the architecture mentioned in the reference paper and we will be using those results as a point of reference.

Future Steps

1. We will take the project forward by incorporating a support vector machine classifier to our deep neural networks model, to perform multi-label classification that will improving the accuracy on the current dataset.
2. We also want to try a Face-Net approach in which we use a triple-loss as the one proposed in Kilian Weinberger's Large Margin Nearest Neighbors. We'll map the images of flowers to a low-dimensional unit hypersphere, so that we may then be able to better learn the inter-class similarity / intra-class differences. This will hopefully also help in improving the accuracy of the flower specie identification, and will make our model robust to variations in image conditions such as lighting.
3. We'll also explore the impact that the choice made from amongst the different pre-processing methods we could use has on the outcome and performance of our model.

Works Cited

[1] M.E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In CVPR, volume 2, pages 1447-1454, New York, 2006.

[2] Liu, Y., Tang, F., Zhou, D., Meng, Y., & Dong, W. (2016, November). Flower classification via convolutional neural network. In *Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA), International Conference on* (pp. 110-116). IEEE.

[3] K.Q.Weinberger,J.Blitzer,andL.K.Saul.Distancemetric learning for large margin nearest neighbor classification. In *NIPS*. MIT Press, 2006. 2, 3

Proposal

Group 5
Wei Zhang
Wenxi Chen

Problem Definition

- You Only Look Once (YOLO) has been thought as the state-of-the-art algorithm for object detection in real time speed.
- YOLOv1 and YOLOv2 both are written in C.
- Although YOLOv1 has been re-coded in Tensorflow, the Tensorflow version of YOLOv2 has not been reproduced.
- This lack leads to compatibility issue to scalable project.

Related Work

- There are two Github repos that have re-coded YOLOv1 into Tensorflow version.
- YOLOv1 paper: https://pjreddie.com/media/files/papers/yolo.pdf
- YOLOv2 project and paper: https://pjreddie.com/darknet/yolo/
- The first one: https://github.com/thtrieu/darkflow
- The second one: https://github.com/gliese581gg/YOLO_tensorflow
- Both are using pre-trained weights to predict.
- The flexibility of adjustment is limited since weights are pre-trained.
- No such existing work is present for YOLOv2.

Preliminary Results

- Tested existing tensorflow implementation of YOLO
  - Limited number of classes
  - Limited accuracy

- Tested the C implementation of YOLOv2 by the author
  - Potential improvement in terms of accuracy can be implemented
  - Problem of mixed bounding boxes on same object should be addressed

Future Plan

1. Implement YOLOv2 with 1000 classes from ImageNet
2. Experiment with different networks to improve the accuracy without significantly sacrificing the speed
3. Address the mixing bounding box issue when large number of classes are incorporated

GIF Upscaling

Developers: GwonJae Cho, YangLu Piao, Anshul Sacheti

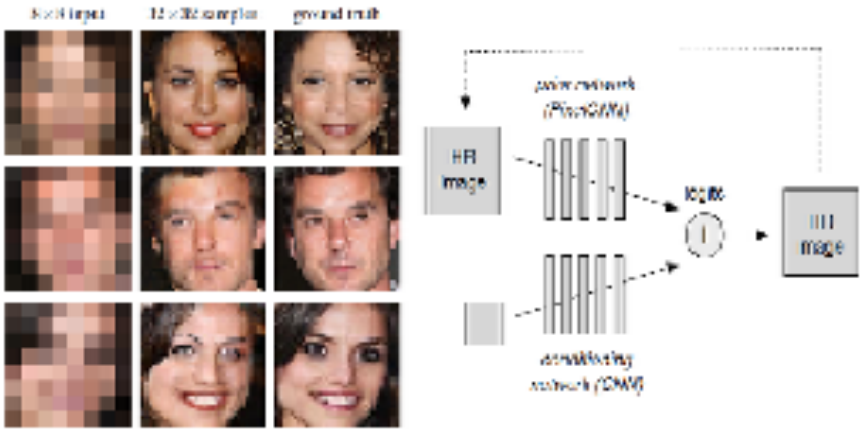
GIF Upscaling Problem

Problem: There are many situations where we cannot afford to capture high resolution videos due to memory/hardware/cost concerns. If we can capture low-resolution videos and upscale them to the relevant resolution then we can bypass these issues.

Project Goal: Take gifs in some domain e.g. faces or dogs and downscale them to 8x8 and 32x32. Subsequently, learn an upscaling network that generates 32x32 gifs from the 8x8 gifs that are indistinguishable to a human observer from gifs naturally at 32x32 resolution.

Related Work

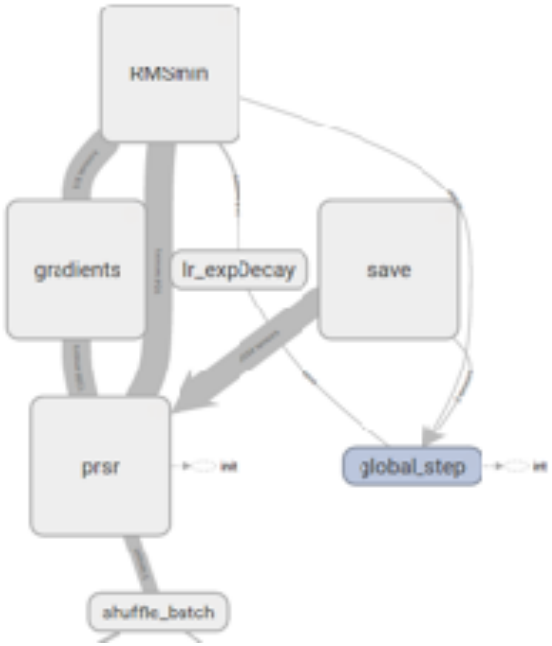
- Linear model - lack of expressivity, often produces blurry frames
- Convolutional Neural Net - Can avoid constructing dictionary. (CNN with MSE Loss[1], ResNet[2] etc.)
- Pixel Recursive Super Resolution
- Conditioning Network
- Prior Network
- PredNet



https://coxlab.github.io/prednet/plots/prednet_animation_with_titles.mp4

[1] C. Dong, C. C. Loy, K. He, and X. Tang. Image superresolution using deep convolutional networks. CoRR, abs/1501.00092, 2012.
[2] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. arXiv:1609.04802, 2016

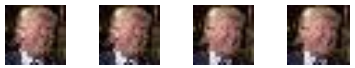
Preliminary Work



On the left is the structure of our current network. And there are two basic blocks contained in “psrs”, one is “conditioning”, the other is “prior”. The former one is utilized to map a low resolution images to a distribution over corresponding high resolution images, while the latter one models high resolution details to make the outputs look more realistic.

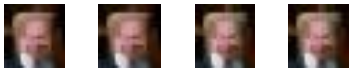
Below are some sample results.

hr_images(Ground Truth):



lr_images:

Gen_images:



Future Plan

Goal: Develop network to better handle changes frame to frame in gifs. Currently each frame is handled individually without any understanding of previous frames. Because

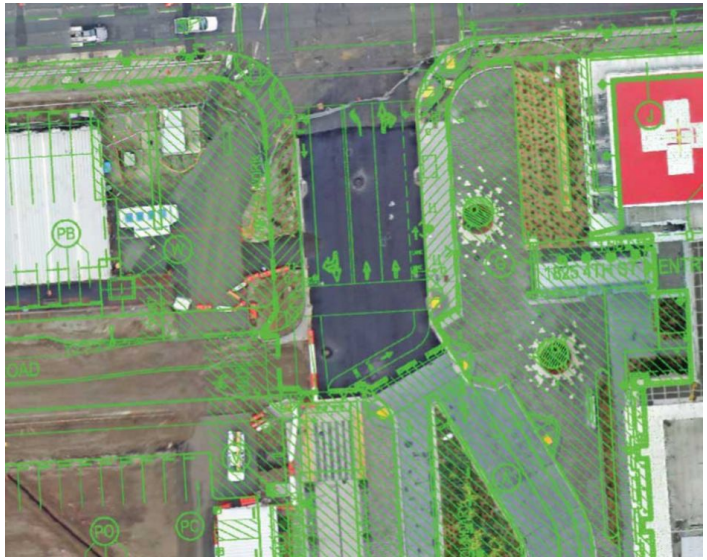
- Use combination of ground truth 32x32 image and generated 32x32 image to derive a more accurate generated image
- Test various RNN layer combinations to determine impact on test frames
- Try an attention model to better track frame to frame changes over time

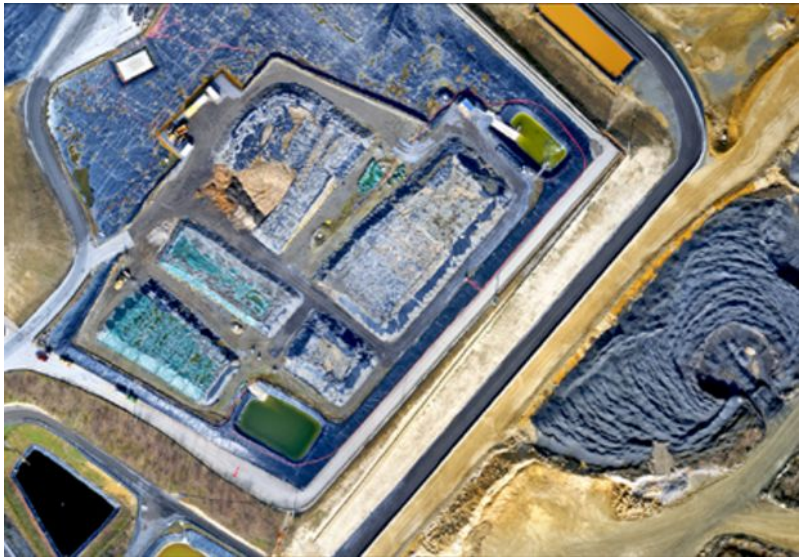
3D Semantic Image Segmentation for Scene Parsing

Mingyang Zheng(mz2594)
Lingyu Zhang(lz2494)
Group 7

Problem to solve

- Pixel-level segmentation based on outdoor 3D point cloud.
- Semantic parsing for 3D outdoor scene.





https://www.skycatch.com/

Related Works

3D semantic segmentation based on outdoor point cloud

- Features and preprocessing methods
  - Pixel color, histogram of oriented gradients, SIFT, BOV, Textons. [BMBM10]
- 3D classification
  - Fixed size of classes. [CS10]
  - A void class was added for classes which were not in the training set. [GRC+08]
- Postprocessing methods
  - Refine a found segmentation and remove obvious errors. [CLP98]
  - Adjust the segmentation to match close edges. [BBMM11]
- Neural network for semantic segmentation
  - Recurrent CNN. [PC13]

[BMBM10] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations"
[CS10] J. Carreira and C. Sminchisescu, "Constrained parametric min-cuts for automatic object segmentation,"
[GRC+08] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller, "Multi-class segmentation with relative location prior,
[CLP98] C. W. Chen, J. Luo, and K. J. Parker, "Image segmentation via adaptive k-mean clustering and knowledge-based morphological operations with biomedical applications,"
[BBMM11] T. Brox, L. Bourdev, S. Maji, and J. Malik, "Object segmentation by alignment of poselet activations to image contours,"
[PC13] P. H. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene parsing,"

Preliminary Results

Dataset:

- EVO3 datasets for ML from Skycatch.

Preparation

- Raw Data Processing (Format)
- SIFT 3D Key Point Extraction
- Parsing Point Cloud into Disjoint Spaces

Future Plan

Parsing Point Cloud into Disjoint Spaces

- Detection incorporating void spaces
  - Detecting the peak-gap-peak pattern
  - Merging
- Canonical Coordinate System Among Spaces

Semantic blocks to pixel level boundary segmentation

- Enforcing Contextual Consistency using CRF
- Updating the Disjoint Space Parsing Results

Group 8

Xiang Hua, Ruixuan Zhang xh2301,rz2364

Our Problem

In the social media, there are a lot of pictures. Sometimes we only remember some "features" of the picture and want to search the image based on that. And people from different countries may search with different language. Our problem is searching a set of images with descriptions in different languages.

Related Work

- *Deep Visual-Semantic Alignments for Generating Image Descriptions*

  https://cs.stanford.edu/people/karpathy/cvpr2015.pdf

  https://github.com/karpathy/neuraltalk2 (github)

- *Faster RCNN: Towards Real- Time Object Detection with Region Proposal Networks*

  https://arxiv.org/pdf/1506.01497.pdf

  https://github.com/rbgirshick/py-faster-rcnn

- *Multilingual Image Description with Neural Sequence Models*

  https://arxiv.org/abs/1510.04709

  https://github.com/elliottd/GroundedTranslation

Preliminary result

- Build environment for Caffe and Keras on Google Cloud
- Paper research about RCNN, fast RCNN, faster RCNN and choose VGG16 as model
- Train model(based on github: neuraltalk2) on MS COCO to generate image descriptions

Future Plan

- We want to train our model with dataset which has more descriptions. And we will have a meeting this week about how to use this data with Pixm.
- After that we will store the descriptions and connection between descriptions and images. After that we will build and train a model which can get image from text.

Project Progress Report

Yu Zheng yz2978
Sun Mao sm4206

Definition of Problem

- Our project name is “from text to song”
- We are focused on utilizing TTS models to relate one specific person’s voice to texts.
- TTS model is an auto-regression model

Related Work

- The state-of-art TTS and music generation model is Deepmind's WaveNet, which uses a casual dilated CNN model with residual blocks to realize the text and audio mapping. The result is quite good but no official code is released.

- Since WaveNet is quite computational expensive(O(2^L), L is the number of hidden layers), there are some paper releasing some work to use dynamic programming method to realize the so-called Fast-WaveNet (O(L)).

- Baidu also gives a novel approach which claims to train the model 400 times faster than WaveNet called DeepVoice this March.

- WaveNet: https://deepmind.com/blog/wavenet-generative-model-raw-audio/
- DeepVoice: http://research.baidu.com/deep-voice-production-quality-text-speech-system-constructed-entirely-deep-neural-networks/

Preliminary Results

- At this point, we have gone through some tensorflow tutorials and tried some basic LSTM models to generate random music based on a database in midi format.

- We've gone through one of the open source reproductions of WaveNet which implements the global conditioned method of WaveNet

Future Plan

- Based on the fact that currently available codes cannot control the content of music we generate, we have to go deeper into the implementation of WaveNet's local conditioning or find a substitution method to finish our project.
- Since the paper did not release too many details about their network, we got contact with some expert in the reproduction of WaveNet with the help of Xiaodong. Hopefully, we can learn more about how this complex network work in the following few weeks.

Group: 10 - Hassan Akbari (ha2436) , Himani Arora (ha2434)
Title: Speech reconstruction from silent video

Goal:

The goal of our project is to reconstruct speech from a silent video of a person speaking. In the past, most of the work has been focused on lip reading, that is, converting silent video frames to text based output. However, in this process a lot of information is lost, for example, the identity of the speaker, the tone of voice and the emotions in it. We are trying to design an end to end deep learning network that directly goes from video to speech.

Previous Work:

We constrain our discussion to lip reading frameworks that employ only neural networks. [1] performed sentence-level sequence prediction for visual speech recognition. Their model which they call LipNet, consists of 3 layers of spatio-temporal CNN + max-pool. The extracted features are then fed into 2 Bidirectional GRUs followed by a linear layer with softmax. It takes as input a sequence of images and outputs a distribution over sequences of tokens. It attained 95.2% sentence-level accuracy on a subset of speakers from GRID database. [2] used an LSTM lipreader that consisted of one feed-forward layer followed by two recurrent LSTM layers, 128 cells each, and a softmax to perform the word classification. Both [1],[2] performed the evaluation on the GRID dataset that has a fixed vocabulary of 51 words with videos shot in a controlled environment. To tackle lip-reading as a real-world problem, [3] used actual in the wild videos (Lip Reading Sentences (LRS) dataset) with unconstrained vocabulary. Their model which consists of three key components: the image encoder Watch, the audio encoder Listen and the character decoder Spell and hence named WLAS, predicts sentences from a talking face video with or without the presence of audio. The Watch module consists of a combination of CNN (based on VGG-M model) and LSTM and the Spell module is similar to an LSTM transducer with added attention mechanism. They achieved state-of-the-art accuracy of 97.0% on GRID and 76.2% on LRS.

In the context of extracting sound features from video, [4] uses a recurrent neural network to synthesize sound from silent videos of people hitting and scratching objects with a drumstick. They computed space-time images for each frame and then performed regular 2D convolutions to extract features based on the AlexNet architecture. This was followed by multiple LSTMs to predict sound features.

We take inspiration from these architectures and propose to design a network that uses some combination of CNN and LSTMs. In contrast to the above described models [1-3] that crop the videos to extract only the mouth regions, we are interested in using the entire face as input to the network. We feel that some information about emotion can be extracted from other parts of the face for example the eyes etc. We train our model on the GRID dataset as the new LRS dataset has not been released yet. However, the GRID dataset has speakers that speak in a monotonic voice that lacks emotion. To compensate for this, we hope to also include another dataset RAVDESS. Due to the small size of this dataset we cannot train the network solely on it.

Dataset:

1. GRID: It is a large audiovisual sentence corpus consisting of high-quality audio and video (facial) recordings of 1000 sentences spoken by each of 34 talkers (18 male, 16 female). Sentences are of the form "put red at G9 now". Each video is 3 seconds long with a sampling rate of 25fps.

2. RAVDESS: Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) is a set of 24 actors (12 male, 12 female) speaking and singing with various emotions, in a North

American English accent. The RAVDESS contains 7,356 high-quality video recordings of emotionally-neutral statements, spoken and sung with a range of emotions. The speech set consists of the 8 emotional expressions: neutral, calm, happy, sad, angry, fearful, surprise, and disgust expressed at two levels of emotional intensity.

Data Preprocessing:

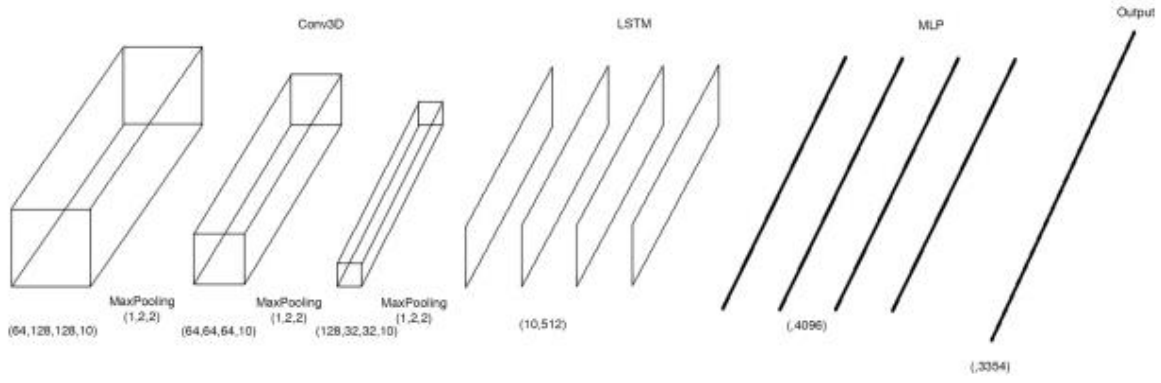
Audio: We first extract audio from the video and reduce its sampling rate to 8k. We then scale the values to lie between -1 to 1 (due to sigmoid activation function) and calculate the spectrogram with 129 frequency bins. We then slice the spectrogram in time to create (overlapping) windowed samples, padding the spectrogram if necessary in order to make the number of audio and video slices consistent.

Video: We convert each frame to grayscale and apply face detection algorithm on the videos to extract only the face region. We then resize each image to 128 x 128 and normalize it to have zero mean and unit standard deviation. We also stabilize the cropped faces to reduce the jittery movement due to pose change of speaker. Similar to audio samples, we slice each video in time to create (overlapping) windowed samples.

Thus, each input to the network is a windowed video sample and the output is a windowed audio spectrogram representative of the actual speech signal. We are currently experimenting with the length of window and overlap. For now, we convert 3 seconds of audio and video to 14 windows with 50% overlap.

Network Architecture:

We started with an end-to-end network that takes the video frames and gives the audio waveform or the spectrogram. A graphical view of the proposed model is depicted in the following figure.

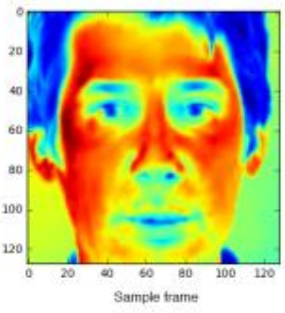


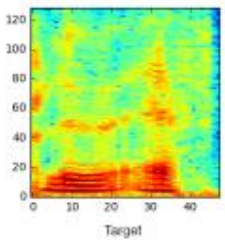
The input to the network is an nd-array of video samples which has the shape, (Samples, Channel, Height, Width, Time). The output could either be the raw (but normalized) audio waveform or vectorized version of its spectrogram. The very first layers (Conv3Ds) try to extract the features from a sequence of frames. Then the features are reshaped and given to four layers of LSTM networks each with the same length as the original sequence to capture the time-dependent features. The extracted spatio-temporal features are then flattened and fed into four layers of feed forward networks and finally, their output is given to an output layer with the same length as the audio waveform (or flattened spectrogram). All the layers have sigmoid nonlinearity, the optimizer is “Adam” with a learning rate of .001 and the loss function is mean squared error.

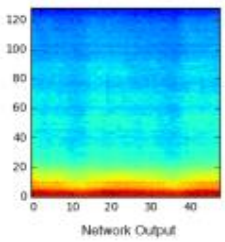
A summary of the network can be seen in the following figure for clarity:

Layer (type)	Output Shape	Param #
conv3d_1 (Conv3D)	(None, 64, 128, 128, 10)	1792
max_pooling3d_1 (MaxPooling3	(None, 64, 64, 64, 10)	0
conv3d_2 (Conv3D)	(None, 64, 64, 64, 10)	110656
max_pooling3d_2 (MaxPooling3	(None, 64, 32, 32, 10)	0
conv3d_3 (Conv3D)	(None, 128, 32, 32, 10)	221312
max_pooling3d_3 (MaxPooling3	(None, 128, 16, 16, 10)	0
conv3d_4 (Conv3D)	(None, 128, 16, 16, 10)	442496
max_pooling3d_4 (MaxPooling3	(None, 128, 8, 8, 10)	0
reshape_1 (Reshape)	(None, 10, 8192)	0
lstm_1 (LSTM)	(None, 10, 512)	17827840
lstm_2 (LSTM)	(None, 10, 512)	2099200
lstm_3 (LSTM)	(None, 10, 512)	2099200
lstm_4 (LSTM)	(None, 10, 512)	2099200
flatten_1 (Flatten)	(None, 5120)	0
dense_1 (Dense)	(None, 4096)	20975616
dense_2 (Dense)	(None, 4096)	16781312
dense_3 (Dense)	(None, 3354)	13741338

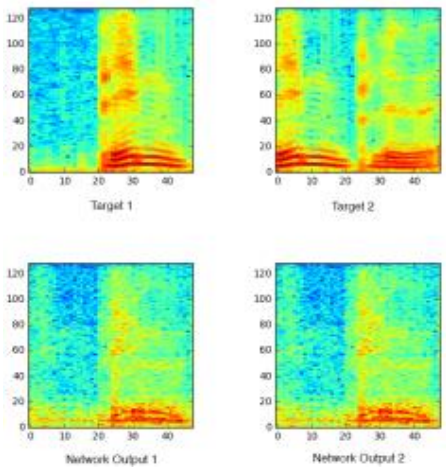
We prepared a sequence of (N_samples, 1, 128, 128, 10) video sequence and (N_samples, 129, 48) audio spectrogram, reshaped the spectrogram to (N_samples, 6192) vectors and trained the network using the mentioned data. Here is the preliminary result that we get with relatively large number of samples:







But, as it can be seen the output is not the same as the target. So, we tried to train the network on a small number of samples to better discuss how it learns the pattern and relation between output and input. Here is a result of training the network on a small number of samples:



As it can be seen, the network output is the same for both inputs (which have different inputs). We tried the same for different inputs and the following conditions:

1. Reshaping image and using Conv2D.
2. Using overlap windows for video and non-overlap for audio.
3. Reducing number of layers for each type to just one layer.
4. Removing MaxPools.
5. Removing regularization and Batch normalization.
6. Removing bias in networks.
7. Removing LSTM layers.
8. Increasing number of features of Conv3D layers.
9. Changing LSTM and MLP nodes numbers.
10. Giving color video instead of grayscales.
11. Changing Batch size, number of epochs, learning rate, regularization parameter, etc.

And got the same result for all the conditions. It's clear that the network thinks all the inputs are the same and tries to take average along all the targets and give the same output for all the inputs. We concluded that since the words utterances are not aggressive in the videos and since we get all the face part and feed it to the network, the network is not able to distinguish differences and changes in each frame. Thus, it either needs a much bigger dataset or more discriminative frames (e.g. taking just the lips area instead of all the face area). As the next steps for improving the results, we are going to train the network on a far bigger dataset, train it on the cropped frames to lips, and finally use pre-trained networks such as ImageNet to directly give discriminative distinct features to the network.

References:

[1] Assael, Yannis M., et al. "LipNet: Sentence-level Lipreading." *arXiv preprint arXiv:1611.01599* (2016).
[2] Wand, Michael, Jan Koutník, and Jürgen Schmidhuber. "Lipreading with long short-term memory." *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016.
[3] Chung, Joon Son, et al. "Lip reading sentences in the wild." *arXiv preprint arXiv:1611.05358* (2016).
[4] Owens, Andrew, et al. "Visually indicated sounds." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

DL FINAL PROJECT REPORT G-11

CHEN-YU YEN /CY2437

YU-CHUN CHIEN / YC3197

THE DEFINITION OF YOUR PROBLEM

- Species identification and counting for wild animal monitoring in camera snapshots
- Count the number of the animals in photo
- Classify animals as different attributes (male/female/child...) and count the amount of each category

RELATED WORK

- Counting objects in images:
  -Learning To Count Objects in Images
  https://www.robots.ox.ac.uk/~vgg/publications/2010/Lempitsky10b/lempitsky10b.pdf
  -CrowdNet: A Deep Convolutional Network for Dense Crowd Counting
  https://arxiv.org/pdf/1608.06197.pdf
  -Learning to Count with CNN Boosting
  http://www.cs.tau.ac.il/~wolf/papers/learning-count-cnn.pdf
- Localization of each sea lion:
  -Mask R-CNN
  https://arxiv.org/pdf/1703.06870.pdf
- Image Segmentation:
  -Segmentation methods in image processing and analysis
  https://www.mathworks.com/discovery/image-segmentation.html

PRELIMINARY RESULTS

Data elaboration:

- Sea lions used to lie on rocks. However, the color of the sea lions and the rocks is similar. Therefore, we may suffer from misidentifying portion of rock as sea lion.
- Baby sea lions are small and often overlap by female sea lion. Therefore, the accuracy rate of counting baby sea lions may be worse.

FUTURE PLAN

- Identify the location of the sea lion
- Extract the image of each sea lion from the photo
- Categorize the extracted images
- Handle the cases of partial occlusion

Deep Learning for Vision, Speech and NLP

Project Report

By: Apoorv Kulshreshtha (ak3963), Samarth Tripathi (st3029)

Project Title: Quora Duplicate Questions Detection

- Quora recently released a corpus of 3 million question pairs with the labels as duplicate or not duplicate. This is the largest paraphrase corpus currently available.
- The challenge is to detect if 2 questions are duplicate, so that Quora can merge the answers to the two separate Qs into one, thereby enhancing user experience.
- Currently, Quora uses random forests approach for this task.
- As this is a classic paraphrase detection problem, and Deep Learning has shown promising results in this field, we aim at exploring the efficiency of various Deep Learning architectures for this task.

Related Work

- [1] is a blog post by engineers at Quora, which provides the problem definition and also discusses some Deep Learning based approaches that can be used to tackle it. We implement their first approach as part of our preliminary result. We plan to implement their second approach as well and continue working upon more improvements.
- [2] is a research paper that presents another advanced approach for the same problem and dataset. It uses bilateral multi-perspective matching (BiMPM) model, with different approaches using attention based stacked LSTM models.
- [3] presents techniques for using CNN based approaches for NLP tasks. We plan on implementing such an approach as well on the dataset to compare between our other approaches.
- [4] tries different vector embedding techniques for the questions and uses those as an input to a stacked Siamese network

[1] https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning

[2] https://arxiv.org/pdf/1702.03814.pdf

[3] http://www.aclweb.org/anthology/D14-1181

[4] http://www.erogol.com/duplicate-question-detection-deep-learning/

Preliminary Results

- For our initial tests, we experimented with the different techniques already explained in the official Quora blog post and a few other blog posts.
- In the subsequent slides, we define the following techniques:
  - LSTM with concatenation from Quora Blog Post (As described at https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning)
  - Converting Qs to vectors using different embeddings and then giving it as input to a stacked Siamese Network to detect if Q pair is duplicate (As described at http://www.erogol.com/duplicate-question-detection-deep-learning/)

LSTM with concatenation from Quora Blog Post

```
Q1 = Sequential()
Q1.add(Embedding(nb_words + 1,
                 EMBEDDING_DIM,
                 weights=[word_embedding_matrix],
                 input_length=MAX_SEQUENCE_LENGTH,
                 trainable=False))
Q1.add(TimeDistributed(Dense(EMBEDDING_DIM, activation='relu')))
Q1.add(Lambda(lambda x: K.max(x, axis=1), output_shape=(EMBEDDING_DIM, )))
Q2 = Sequential()
Q2.add(Embedding(nb_words + 1,
                 EMBEDDING_DIM,
                 weights=[word_embedding_matrix],
                 input_length=MAX_SEQUENCE_LENGTH,
                 trainable=False))
Q2.add(TimeDistributed(Dense(EMBEDDING_DIM, activation='relu')))
Q2.add(Lambda(lambda x: K.max(x, axis=1), output_shape=(EMBEDDING_DIM, )))
model = Sequential()
model.add(Merge([Q1, Q2], mode='concat'))
model.add(BatchNormalization())
model.add(Dense(200, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(200, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(200, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(200, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy', 'precision', 'recall', 'fbeta_score'])
```

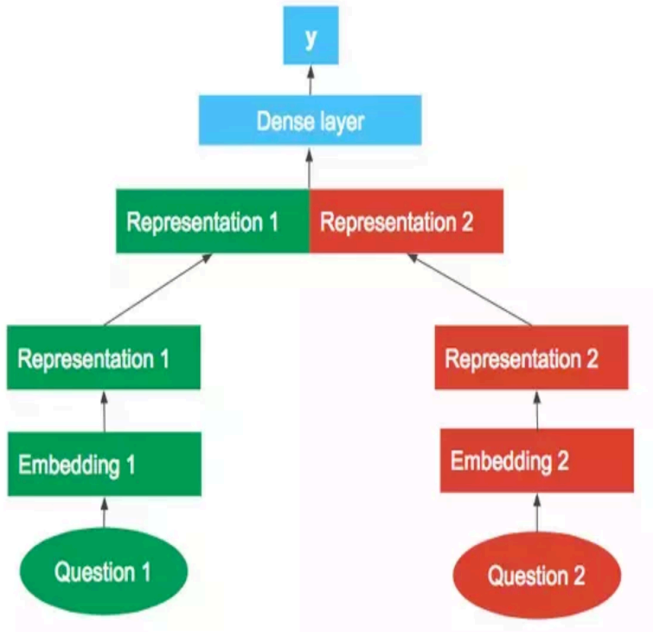


Figure 1: Architecture of approach 1, "LSTM with concatenation"

Results from LSTM with concatenation

We were able to achieve 0.82 accuracy on validation set (40,000 samples) and 0.86 on test set (4000 samples)

```
Starting training at 2017-04-05 10:57:56.339776
Train on 360000 samples, validate on 40000 samples
Epoch 1/25
806s - loss: 0.4446 - acc: 0.7843 - precision: 0.7212 - recall: 0.7001 - fbeta_score: 0.6988 - val_loss: 0.4119 - val
_acc: 0.7994 - val_precision: 0.7074 - val_recall: 0.7328 - val_fbeta_score: 0.7096
Epoch 2/25
807s - loss: 0.4103 - acc: 0.8045 - precision: 0.7438 - recall: 0.7370 - fbeta_score: 0.7294 - val_loss: 0.3985 - val
_acc: 0.8115 - val_precision: 0.7218 - val_recall: 0.7549 - val_fbeta_score: 0.7285
Epoch 3/25
807s - loss: 0.3833 - acc: 0.8203 - precision: 0.7610 - recall: 0.7650 - fbeta_score: 0.7527 - val_loss: 0.3878 - val
_acc: 0.8160 - val_precision: 0.7174 - val_recall: 0.7841 - val_fbeta_score: 0.7404
Epoch 4/25
807s - loss: 0.3616 - acc: 0.8326 - precision: 0.7742 - recall: 0.7870 - fbeta_score: 0.7706 - val_loss: 0.3799 - val
_acc: 0.8220 - val_precision: 0.7283 - val_recall: 0.7846 - val_fbeta_score: 0.7463
Epoch 5/25
806s - loss: 0.3433 - acc: 0.8435 - precision: 0.7867 - recall: 0.8028 - fbeta_score: 0.7855 - val_loss: 0.3827 - val
_acc: 0.8218 - val_precision: 0.7299 - val_recall: 0.7812 - val_fbeta_score: 0.7459
Epoch 6/25
824s - loss: 0.3281 - acc: 0.8526 - precision: 0.7972 - recall: 0.8170 - fbeta_score: 0.7982 - val_loss: 0.3764 - val
_acc: 0.8275 - val_precision: 0.7365 - val_recall: 0.7901 - val_fbeta_score: 0.7534
Epoch 7/25
802s - loss: 0.3148 - acc: 0.8591 - precision: 0.8060 - recall: 0.8249 - fbeta_score: 0.8069 - val_loss: 0.3831 - val
_acc: 0.8266 - val_precision: 0.7455 - val_recall: 0.7652 - val_fbeta_score: 0.7465
Epoch 8/25
803s - loss: 0.3048 - acc: 0.8651 - precision: 0.8137 - recall: 0.8324 - fbeta_score: 0.8146 - val_loss: 0.3870 - val
_acc: 0.8245 - val_precision: 0.7621 - val_recall: 0.7237 - val_fbeta_score: 0.7324
Epoch 9/25
```

Results from using different embeddings with Siamese Network

- Model 1 with Glove embeddings:
  - Accuracy on test set: 56.88%
- Model 2 with tf-idf vectors:
  - Accuracy on test set: 75.62%
- Model 3 with tf-idf vectors and normalized train data:
  - Accuracy on test set: 74.90%
- Model 4 with word2vec embeddings:
  - Accuracy on test set: 67.56%

Future Plan

1. A simple extension to our preliminary approach is using more variations on combining encodings received from LSTM layers, to be processed later with Dense FC layers. The current approach proceeds to simply concatenate the encodings. [1] suggests using difference and angle vectors instead, as Illustrated in adjacent Figure

2. Another approach is to treat sentences with word embeddings as images, stacking the two question images on top of each other, padding to ensure same dimensionality, and running a CNN based classification like [3].

3. Other approaches could be using minor variations of the architectures discussed, variations in activation functions and trying ensemble learning based model.

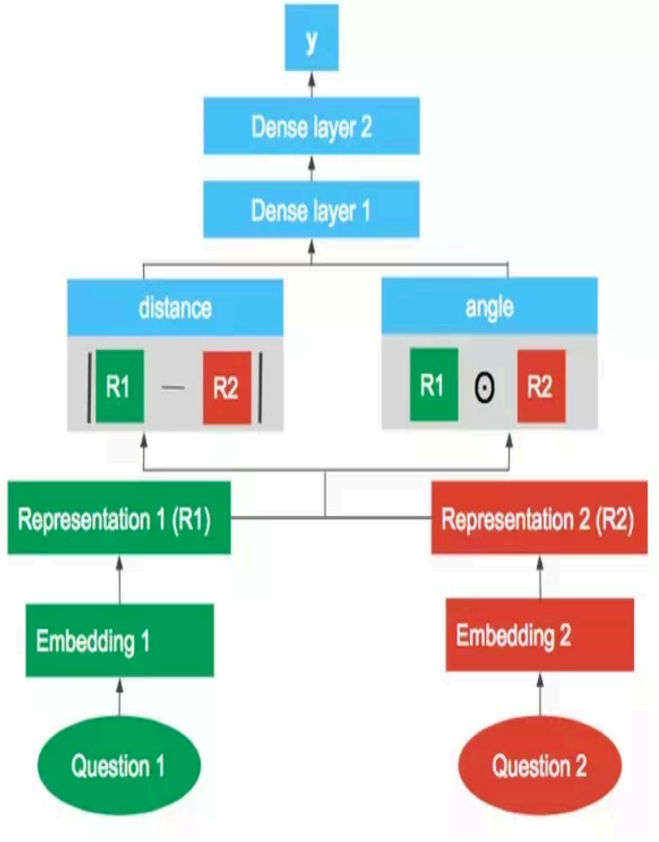


Figure 2: Architecture of approach 2, "LSTM with distance and angle"

Language Modeling for Large Vocabularies

Harish Shanker (hrs2139)
Akshay Khatri (ajk2237)

Problem

- We are suggesting a new method for NLP tasks like phrase completion that will be more robust for large vocabularies
- Can we predict Word Vectors instead of predicting the words themselves?
- For large vocabularies, predicting the words themselves is very difficult / impossible since we need |V| nodes in the softmax layer of the network
- By predicting word vectors, the model will take up much less space thereby making it feasible to predict over a large vocabulary
- This idea can be extended to any general problem where we have distributed representations available for the data, but we will keep it limited to word embeddings and NLP for this project

Related Work

- DeVISE
  - Predicts word embeddings instead of probabilities from a softmax layer while doing ImageNet classification.
- Exploring the Limits of Language Modeling [2016]
  - Extends current models to deal with corpora and vocabulary sizes
  - Introduce techniques such as character Convolutional Neural Networks and Long-Short Term Memory on the One Billion Word Benchmark
- Recurrent neural network based language model [2010]
  - Provides the key idea to use RNNs for Language modelling.

Preliminary Results

- In order to test out whether it is possible to predict word vectors directly, we are evaluating on language modelling
- For language modeling, we need the model to output probabilities for words
- We do so by having the model predict a probability distribution over all word vectors and train it to predict these parameters
- Specifically, the model predicts the parameters for a Gaussian distribution, mean and variance
- We train the model to simply maximize the probability of the correct labels. We are still in the training process and have no concrete results at this time
- Data: One Billion Word Benchmark
    - http://static.googleusercontent.com/media/research.google.com/pt-BR//pubs/archive/41880.pdf

Future Plans

We have the basic architecture to test this idea. But, most likely it won't work in the first attempt. Here are a few ideas to mitigate some of the problems that may arise:

- Modify the loss function to be more informative for what the network is predicting
- Modify the architecture to make each dimension of the produced embedding to be more independent of each other. This will mean a normal fully connected layer will not work here. We might need to treat each dimension as a separate regression problem.
- Retrain the word embeddings so that they are more suitable for this task. This might mean learning embeddings with a deep net rather than a shallow one.

SHORT PROJECT REPORT

Zixiaofan Yang, Xing Lan

Problem Definition

Task:

- Emotion Recognition in Speech

Dataset:

- RECOLA database
  - 46 participants, 5 minutes each
  - 6 annotators measured emotion continuously
  - Two dimensions: arousal and valence

Proposed neural network structure:

- Apply CNN layers on both raw signals and MFSC features
- Feed the output of the CNN layers into two Bi-LSTM layers
- Target output of the Bi-LSTM layers: arousal and valence score

Related Work

- [1] Trigeorgis, George, et al. "Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network." Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on. IEEE, 2016.
- [2] Abdel-Hamid, Ossama, et al. "Convolutional neural networks for speech recognition." IEEE/ACM Transactions on audio, speech, and language processing 22.10 (2014): 1533-1545.
- [3]Ringeval, Fabien, et al. "Introducing the RECOLA multimodal corpus of remote collaborative and affective interactions." Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on. IEEE, 2013.

Preliminary Results

- Implemented the structure in Trigeorgis et al.[1]
- Tuned parameters
- With smaller dataset size and layer size, we obtained similar results with the paper
- Performances are measured by concordance correlation coefficient:

Model	Arousal	Valence
Trigeorgis et al	**0.686**	0.261
Our model	0.679	**0.321**

Future plan

- Compute MFSC/MFCC features
- Organize the features into CNN layers
- Integrate the CNN output from the raw signals and MFSC features into Bi-LSTM layers
- Experiment with different layer/kernel sizes

GENERATIVE ADVERSARIAL NETWORKS FROM AUDIO TO IMAGE

Deep Learning for Computer Vision, Speech and Language

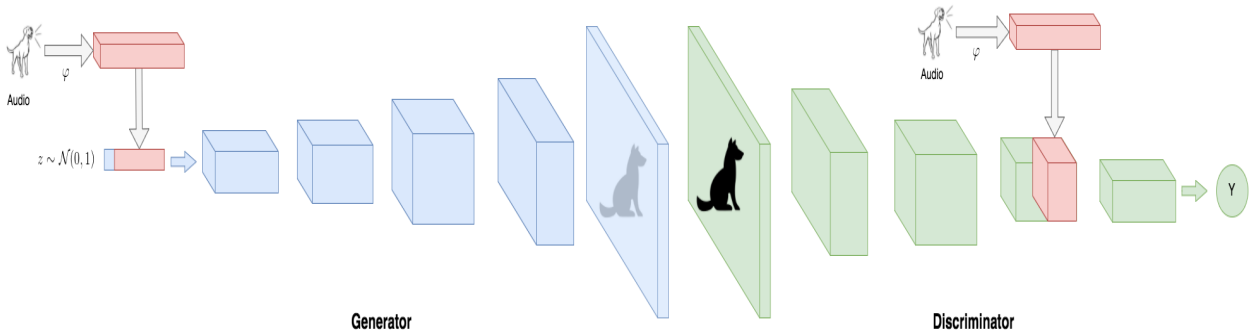
Group 15

Jose Vicente Ruiz Cepeda

Pablo Vicente Juan

PROBLEM DEFINITION

- The aim of the project it to develop a deep convolutional generative adversarial network (DCGAN) to bridge the advances in audio and image modeling. To the best of our knowledge, this is the first DCGAN that aims to generate images based on sounds
- The generator receives a sound and generates an image that can be identified with such a sound, i.e. given the sound of a bark, the aim is to generate the image of a dog. The discriminator have to determine whether the new image comes from the real distribution or from the generator



RELATED WORK

- GANS, developed by Goodfellow *et al.,* make possible to generate new samples by learning the data distribution using the minimax algorithm, where both players try to perform the best possible move. His research is the based of any GAN developed and represents a breakthrough in generative models
- In their work, Radford *et al.* extend Goodfellow´s research by combining a deep convolutional network with a GAN model, which led to the development of a convolutional model able to generate images from a given set. This model is the baseline for any generative adversarial approach that aims to create new images
- Reed *et al.* devised a DCGAN to generate images given a sentence. In this case, the sentence explicitly describes what the network has to generate leaving limited space for improvisation. Our model is inspired in their idea of encoding the conditional information before passing it to the network. However, our conditional model provides the freedom to depict anything that resembles to such a sound.

PRELIMINARY WORK

- A DCGAN without the conditional audio architecture has been trained in the following datasets in order to learn to train this unstable neural network:
  - Stanford Cars Dataset provides a relatively uniform dataset which might be beneficial for a naïve network. In this case, most images have similar shapes and only colours are altered
  - Oxford Buildings Dataset challenges our current network due to the variety of shapes involved that can make the network collapse before obtaining any relevant results

Image quality evolution of the generator

FUTURE PLAN

- The following steps can be divided in three:
  - Building a dataset that combines both images and relevant sounds. If no dataset with such features is found, a new one will be build. At the moment, we have gathered different datasets that includes animals, cars or scenes. We would only need to combine these images with sounds using sources such as the AudioSet released by Google
  - Extending the architecture with the conditional audio layer. Our main challenge consists on finding a suitable encoder that maintains all audio features at the same time that allows the concatenation with an existing layer
  - Testing the network capabilities to generate images conditioned on audio. Training DCGANs is a particularly complicated task due to their instability, they tend to collapse with no reason. Tuning the parameters will be a trial and error routine where best practices will be applied to speed up the process

REFERENCES

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NIPS*, 2014.
- Radford, A., Metz, L., and Chintala, S. Unsupervised rep- resentation learning with deep convolutional generative adversarial networks. 2016.
- S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text-to-image synthesis. In ICML, 2016b.

Group 16

Abhishek Jindal, Siddarth Varia, Oscar Chang aj2708, sv2504, oc2241

Problem Definition

- Determine if two questions on the Quora Question Similarity dataset are asking questions with similar intent
- This is related to the general problem of semantic similarity between two sentences, but applied specifically to the case of erotetic semantics

Related Work

- People have tried employing Siamese LSTMs for testing semantic similarity between sentences
- We think it’s a better approach to concatenate two sentences and train a network on the concatenation, because we wouldn’t have a good loss function for a Siamese model given that the Quora dataset has binary labels for whether two sentences are similar

Preliminary Results

- We achieved an accuracy of 0.84 on a validation set after 200 epochs on training on a deep net powered by standard LSTMs

Future Plan

- We intend to try out data augmentation and attention-based LSTMs amongst other techniques to further boost our accuracy on this task

Crabgrass Classification

Manu Gandham & Jonathan Koss

Data Annotation

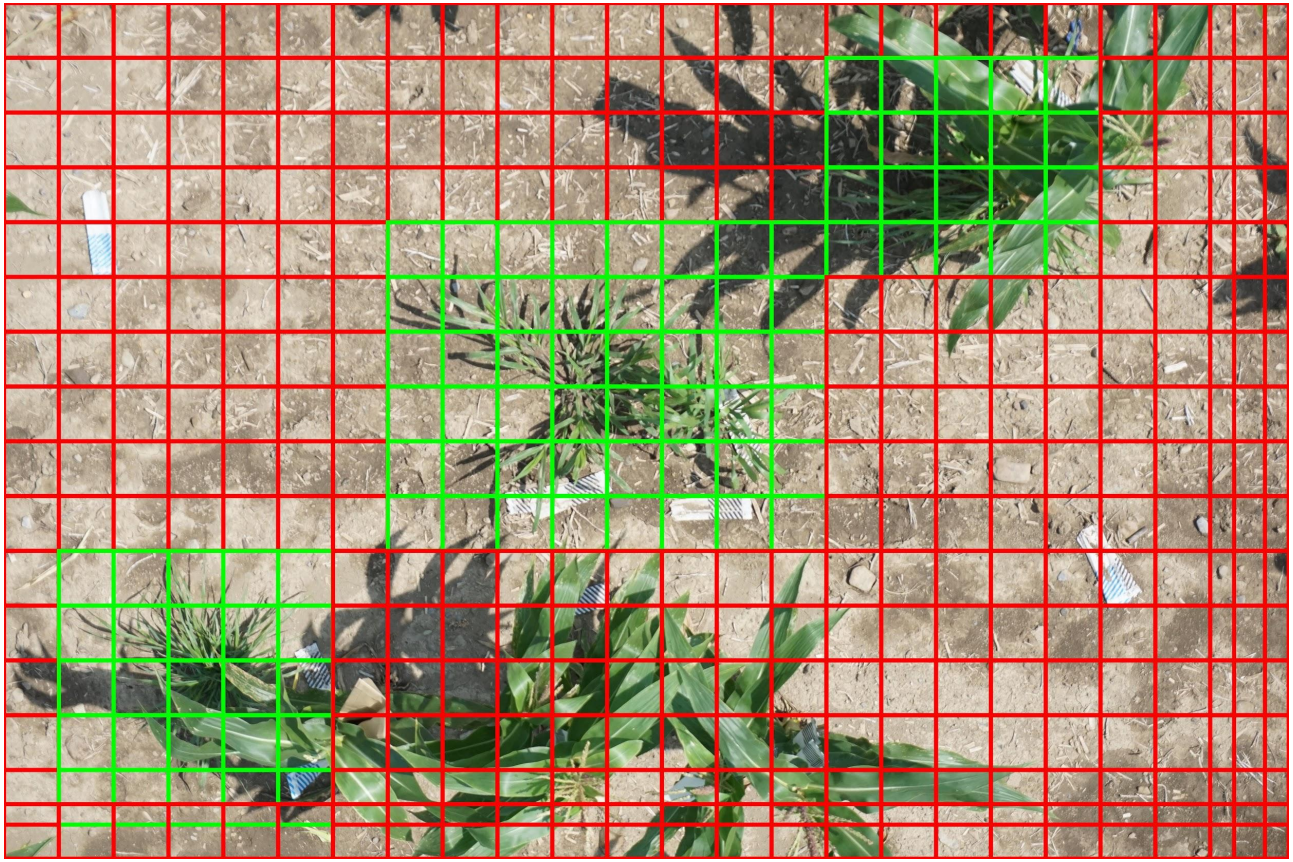
We have a large dataset of images sourced through MTurk

Data Labeling

We have selected the regions with the greatest overlap to serve as labeled examples of weeds

Data Segmentation

We separated each image into a grid of images, each with its own label



Final Dataset Specs:

198,198 Images

Each image is 224x224x3(RGB)

Class breakdown: Each of these images is given a binary label if there are weeds (0->no weeds, 1-> weeds)

Training

We are currently training a network on the task of classifying an image of the cornfield as containing or not containing weeds. Our biggest challenge so far has been manipulating and working with the data since there are so many images.

We are using a replica of the vgg-16 network with weights pre-trained on imagenet

We have also replicated the darknet-19 network from the paper YOLO9000 in keras including the necessary modifications to make it compatible with our dataset and question we are trying to answer.

Next Steps

We expect our trained network to have a baseline accuracy of 91%, but due to the high amount of mislabeled training samples we expect to encounter some issues

We can generate a new dataset with more accurate labels which score each tile image from 0-3 depending on how many MTurk users classified the tile as a weed. Currently we're only selected tiles with a score of 3 to use as the '1' in our binary classification task, but by expanding the output space to include lower scores we will get a more complete representation of our ground truth.