EmotiGAN: Emoji Art using Generative Adversarial Networks

Marcel Puyat

***Abstract*— We investigate a Generative Adversarial Network (GAN) approach to generating emojis from text. We focus on two interesting research areas related to GANs: training stability and mode collapse. In doing so, we explore a novel GAN training approach that involves training a generator with different images with the same conditional label in order to produce more varying kinds of images with Conditional GANs.**

I. Introduction

In this work, we are interested in translating text in the form of a phrase or "search query" directly into emoji image pixels. One might imagine an application for this wherein a user inputs a short search query and is given a unique emoji relating to their search terms.

Similar recent works[1] have tackled the task of translating text into images by solving two sub-problems: first, learn a text feature representation, or embedding, that captures the important visual details of the domain; and second, use these features as input into a generative model to generate realistic images that a human might mistake for real.

EmotiGAN solves this problem by reusing works from 2 different areas. First, we reuse most of the architecture of Deep Convolutional GANs[2] (DCGAN) which has shown to do quite well on a broad range of domains related to image generation. We then use a pre-trained *word2vec*[3] model to convert text into an embedding that we can use as input into our GAN. Section 3 covers more details on the design and implementation of these two components.

Along the way, we make some alterations to the DCGAN architecture in order to stabilize training and produce more varying generated images given the same text label.

Fig. 1: Examples of EmotiGAN generated emojis from phrases. The phrase above each emoji was converted into an embedding using *word2vec* and then fed as input into a GAN.

II. Related Work

A. Generative Adversarial Networks

A Generative Adversarial Network[4] consists of two neural networks, a generator and discriminator, whose cost functions set up a minimax game wherein the discriminator learns to classify images as real or fake, and the generator attempts to produce realistic synthetic images in order to fool the discriminator.

More formally, the generator and discriminator play the following minimax game with value function $V(D, G)$:

$$\min_G \max_D V(D,G) = log(D(\mathbf{x})) + log(1 - D(G(\mathbf{z})))$$

Where $\mathbf{x}$ is a real image, $\mathbf{z}$ is a noise variable drawn from some prior, D attempts outputs 1 or 0 given a

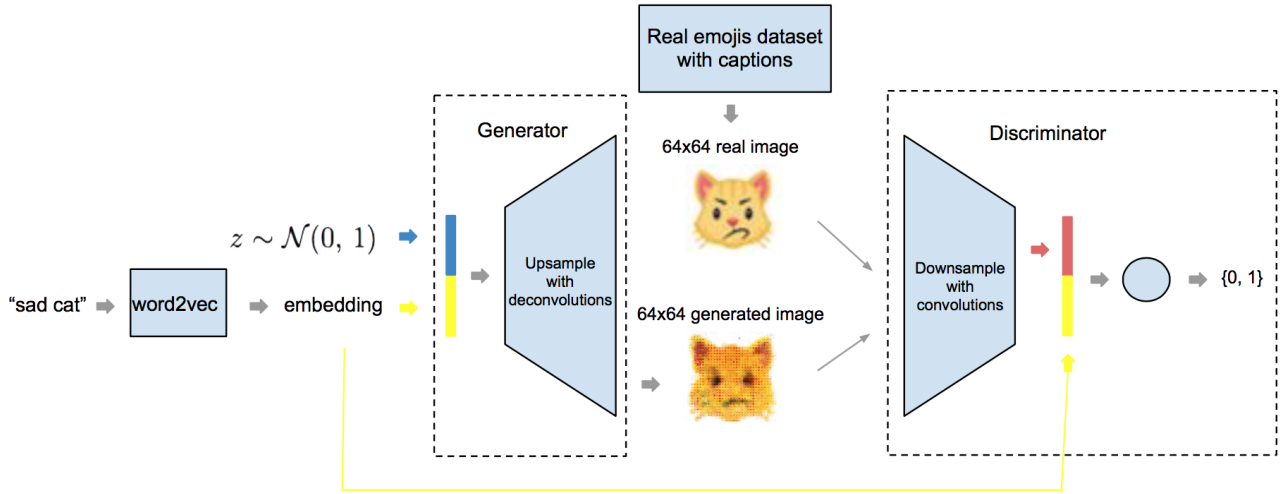

Fig. 2: EmotiGAN high level architecture

real or fake image respectively, and G attempts to produce a realistic image given the noise variable **z**.

GANs are notoriously difficult to train to convergence, wherein the discriminator and generator learn at a similar pace where neither one ends up overpowering the other, leading to either exploding or vanishing gradients for the other. Many heuristics and formal techniques[5] have been explored in order to improve training stability for GANs, and we explore many of these techniques in order to stabilize EmotiGAN. Another GAN failure mode that has not been fully solved is the issue of mode collapse, where the generator outputs images with little or no variety, despite being trained on a dataset with high variance of image types.

There are no clear evaluation metrics used in the field of unsupervised image generation. Goodfellow et al.[5] suggest an *inception score* metric that we explore using in this work in order to provide quantitative results in addition to the qualitative analysis done by simply examining the synthetic images.

B. Conditional GAN

Mirza et al.[6] explored adding an additional input variable to both the discriminator and generator to allow for more control over the generated images. In EmotiGAN, for example, this additional conditional input is the vector-representation of the text input.

The issue of mode collapse (wherein given the same conditional input c, the generator outputs the exact same image) with Conditional GANs is mostly unsolved, as many[s] have found that the generator learns to ignore the noise variable **z** and only conditions on the conditional input c.

C. Text to Image Synthesis

Zhang et al.[1] were able to train a text-to-image GAN on a dataset of bird images. Their application differs from this work in that they were interested in translating full sentences into images. In order to do so, they trained a word embedding model intended to capture the visual semantics behind a given sentence related to birds. Our work simply uses a pre-trained *word2vec* model[3], but one can imagine possible extensions to this work wherein a word embedding model is trained especially for emojis.

III. EmotiGAN

EmotiGAN draws inspiration from the DCGAN architecture, using a similar approach with an upsampling ConvNet used for the generator and a downsampling ConvNet used as the discriminator. Figure 2 gives a high level view of the overall architecture. Here, we lay out the unique aspects of EmotiGAN:

A. Heuristic Cost Function

The generator's cost function in the original GAN paper[4] directly minimizes the negative of the discriminator's cost function:

Fig. 3: Images generated when using text input: "tongue"
Left: Images with our mode collapse training mechanism.
Right: Images without our mode collapse training mechanism

$$J(G) = log(1 - D(G(\mathbf{z}))$$

However, when training with this objective, we found that some training attempts resulted in early vanishing gradients for the generator that would never recover. In the scenario that the discriminator happens to get off to an "early lead" and starts rejecting the generator's outputs, the generator's objective function quickly goes to zero.

Goodfellow et al.[5] found that a heuristic objective function wherein the generator instead tries to maximize the number of synthetic images classified as real:

$$J(G) = -log(D(G(z))$$

We employed this heuristic objective function and found a significant reduction in the number of training attempts that would fail early. Notice that with this objective function, the generator's loss increases as the discriminator starts rejecting more of its outputs, leading to bigger gradients when it is doing poorly. In other results, Goodfellow et al.[7] explored the possible downsides of this heuristic cost function and found that it did not significantly affect the long term convergence of GAN training.

B. Addressing mode collapse

When attempting a first pass at training emoji image generation without conditioning on text labels, we found that, as commonly seen in most GAN training attempts, the generator had very little variance in its outputs. Goodfellow et al.[5] suggest using a technique they called *minibatch discrimination* wherein the discriminator examines all images in a given batch and measures similarity between images. This forces the generator to achieve similar variance levels as real batches of images.

However, as mentioned in our Related Works section, Conditional GANs suffer from a slightly different form of mode collapse that is not rectifiable with vanilla minibatch discrimination: a given batch of images with different conditional labels might look different by virtue of the discriminator requiring an image to resemble its conditional input, but images with the same conditional input tend to look the same.

We employ a novel training mechanism to address this issue. Instead of simply running usual training iterations wherein the conditional inputs are drawn in random batches, we alternate random batches with batches that all contain the same label. We then combine this with Goodfellow's minibatch discrimination, and the result is that the generator is forced to use the noise its input noise variable $\mathbf{z}$ in order to vary its outputs with the same conditional label and not get rejected by the discriminator.

IV. Dataset

We trained on a dataset of 2,000 emoji images scraped from *unicode.org*[8], training on Tensorflow on Paperspace Cloud GPUs (NVIDIA Quadro P6000). Examples of dataset emojis are included in the appendix. One important step we took on the data to stabilize GAN training is to add instance noise to the discriminator's input in order to stabilize training early in the GAN minimax game. Sonderby et al.[9] found that adding noise to the discriminator input ensures that even in

the early phases of training when the generator is outputting nonsensical images, there will be some overlap between the real images with noise and the generator outputs with noise. This overlap is crucial to the stabilization of GAN training, which implicitly relies on the assumption that the KL-Divergence between the generated distribution and real data distribution. When there is zero overlap between the real data and generator output, the KL-Divergence is undefined and this leads to irrecoverable issues with training.

V. Experimental Results

Figures 1 & 3 show some of our qualitative results. Unsupervised Generative Models do not have a standard quantitative metric that is used. Nonetheless, we attempt to use a modified version of Goodfellow's inception score metric mentioned earlier, wherein we use a modified version of our trained discriminator (with a softmax layer at the end) in order to the output class label probabilities that we need to measure the inception score. His proposed formula is:

$$exp(KL(p(y|x)||p(y)))$$

Which is the KL divergence between the probability that a classifier running the Inception model trained on your dataset will assign the correct label to a generated image (which should be higher the better a GAN is), and the probability of a given class appearing $p(y)$ should have high entropy because we desire varied outputs. Because we don't have a pretrained Inception model on our emoji dataset, we instead converted our Discriminator model into a CNN classifier with a softmax layer at the end. This allowed us to compute $p(y|x)$ and, by marginalizing over our generated images, $p(y)$. We found that this modified inception score when training with our proposed method of addressing mode collapse is notably higher due to the greater variance in the generator's outputs. The following table displays our scores when training with our proposed training mechanism meant to vary the outputs more by using batches with varying outputs and the same label, and without it.

While we took some steps to avoid overfitting our training set, such as dropout both the discriminator and generator model, adding noise to the inputs to both models, and adding noise to the labels by having the discriminator not always measure it's cost against exactly 1 for real images or exactly 0 for fake images, we do think that there was some overfitting that occurred around how the generator uses the conditional input. More specifically, when we provided the generator with words that deviated a good amount from the words in the training set, many of the emojis it generated were nonsensical. We've included some of these in the appendix and believe that this can be alleviated with: A.) A larger, more varied dataset, and B.) a word embedding space that makes more sense to use for emojis.

Inception Score

Same labels in batches	Without same labels in batches
8.4	4.9

VI. Conclusion

In conclusion, we were able to train a model to generate emojis similar to those in our training set while discovering a novel training mechanism that is able to have a Conditional GAN generate more varied kinds of outputs. This is an open research problem and the hope is that this work can make a contribution to these types of problems. Future work should consider trying to scale our training mechanism to larger datasets better (possibly by only replicating N training examples with the same label in a given batch), using a more sensible word embedding for the domain of emojis as opposed to Google News word2vec, and using a larger dataset for training in order to generalize to more varying kinds of words when generating emojis from text.

References

[1] Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., and Metaxas, D. (2016). Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv preprint arXiv:1612.03242.

[2] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013. https://code.google.com/archive/p/word2vec/

[4] Goodfellow, Ian J., Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron C., and Bengio, Yoshua. Generative adversarial nets. NIPS, 2014.

[5]] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. arXiv preprint arXiv:1606.03498, 2016.

[6] M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
[7] William Fedus, Mihaela Rosca, Balaji Lakshminarayanan, Andrew M. Dai, Shakir Mohamed, Ian Goodfellow. Many Paths to Equilibrium: GANs Do Not Need to Decrease a Divergence At Every Step
[8] https://unicode.org/emoji/charts/full-emoji-list.html
[9] C. K. Sonderby, J. Caballero, L. Theis, W. Shi, F. Huszar. Amortised MAP Inference for Image Super-resolution

VII. Appendix

A. *Training data examples*

Fig. 4: ”Smiling face with halo”

Fig. 5: ”Smiling Devil Face”

Fig. 6: ”Pile of poo”

B. *More generated emojis*

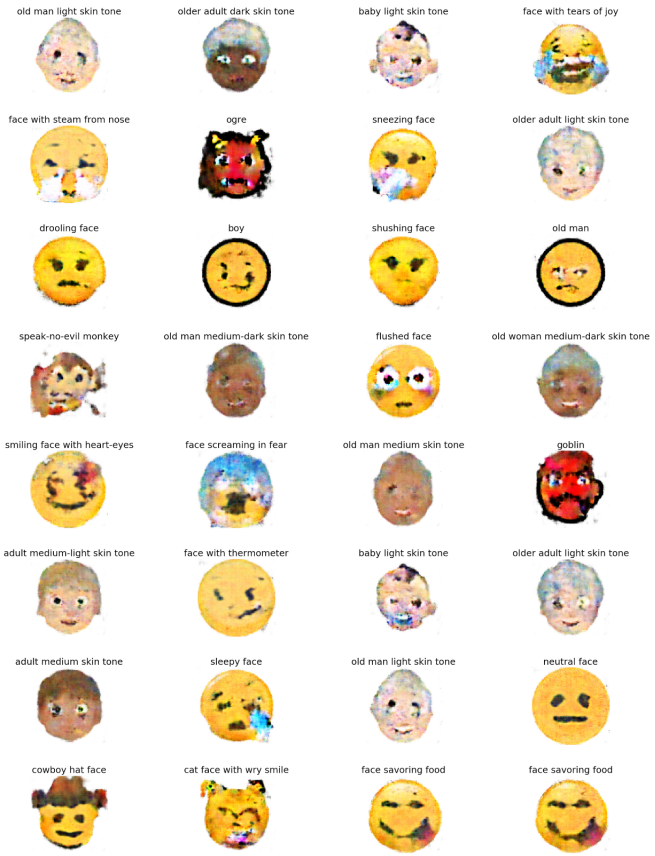

Fig. 7: Various different labels

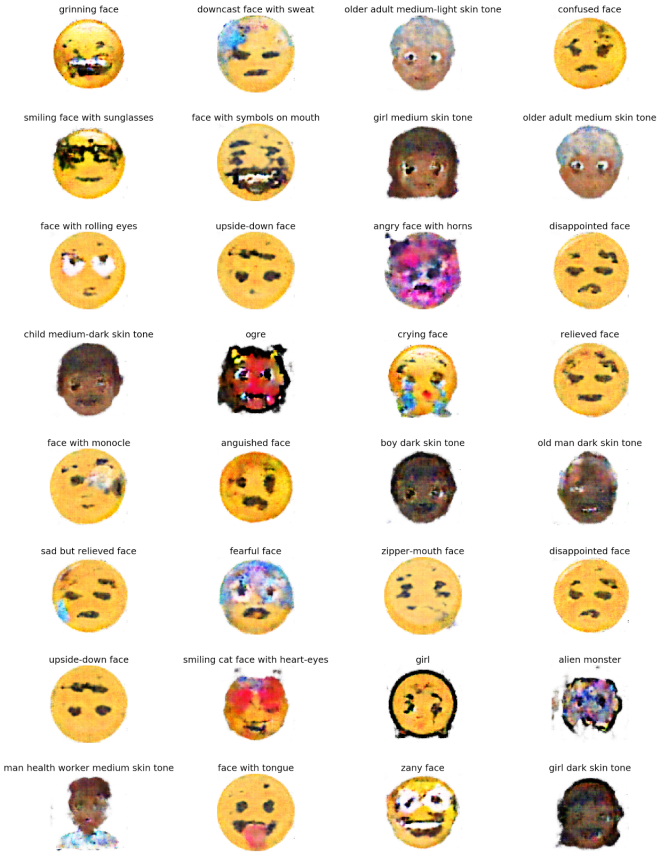

Fig. 8: Various different labels, 2

Fig. 9: Input text: ”glasses”

Fig. 10: Input text: ”glasses”, without our output-varying training mechanism