Automatically Review Tinder Profiles with FaceNet and Machine Learning

DSI Spring Symposium 2019

Charles Jekel

March 31, 2019

University of Florida
PhD candidate in the MAE department
cjekel@ufl.edu
`https://jekel.me`

A lot of people are using online dating

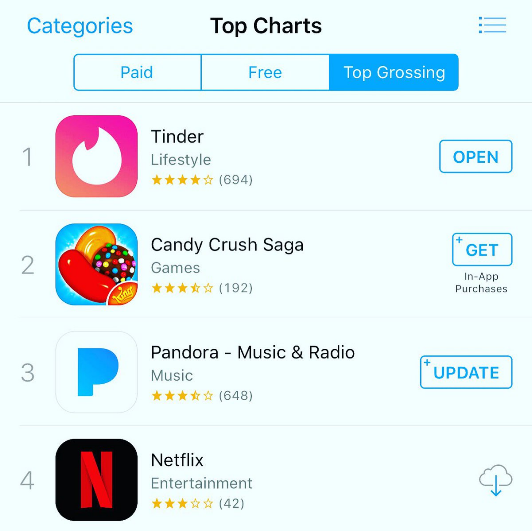
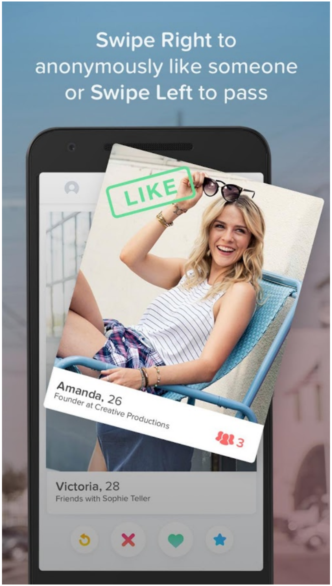


Figure 1: In September of 2017, Tinder, the online dating application became the top grossing app in the iOS store.

What is Tinder?

- location based online dating
- presented with singles near you
- view profiles one at a time
- like or dislike the profile

Tinder sends a notification when two people like each other, where they can then message each other.



The problem

It takes a lot of time to review profiles.

Tinder has a lot of data, and could very well build a model to automatically like profiles for me.

Why can't I just liked everyone?

- Free users get 100 likes a day
- For $10 a month you can get unlimited likes
- You will be penalized by Tinder for not being *selective*
- Liking all of the profile ruins the Tinder atmosphere
- Still need to filter unnecessary matches

So it became clear to me

I could use machine learning to automatically review Tinder profiles.

- I had no idea about how to go about doing this
- But I knew that I would need a lot of data
- So I built a custom application to interface with Tinder
- Which created a dataset from all of the profiles that I reviewed

My dataset of reviewed Tinder profiles at a glance

- 8,545 profiles reviewed
- 2,411 profiles liked
- 38,218 profile images
- 640x640 pixel RGB images
- 1.9 GB of data

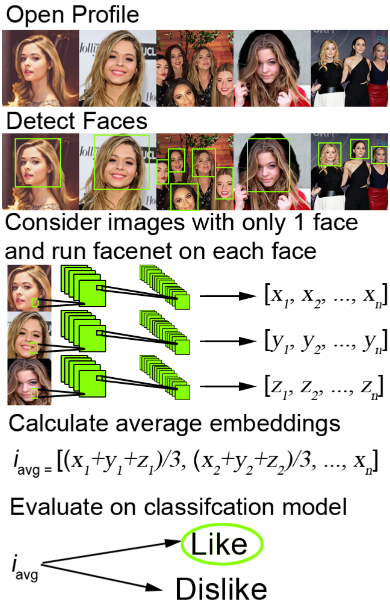
No low hanging fruit

I tried everything and couldn't find a pattern in my dataset.

Where the profiles I liked random?

Could state-of-the-art techniques in Facial Classification help?

How I ended up finding a pattern in the data



Enter FaceNet - A unified embedding for Face Recognition

A 7.5 million parameter neural network of 1.6 GFLOPS to apply facial classification at scale [3].

- Set a Labled Faces in the Wild (LFW) [2] record with 99.63% accuracy
- Turns face into a vector of features

How does FaceNet work? [3]

- L_2 norm distance between images of the same face are small
- L_2 norm distance between images of different faces are large
- Triplet loss function minimizes the distance between an anchor and a positive, while maximizing the distance between the anchor and a negative
- Convolutional neural network (CNN)

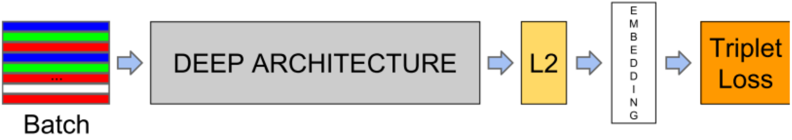


Figure 2: Model structure figure from [3].

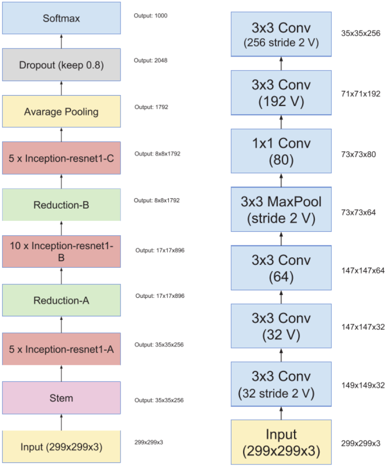


Figure 3: ResNet v1 Architecture model weights are a 200mb file. Figure taken from [4].

FaceNet implementation at a glance

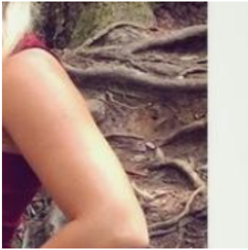
- Created by David Sandberg
  `https://github.com/davidsandberg/facenet`
- MIT license
- Python + TensorFlow
- Multitask Cascaded CNN for face box detection [6]
- Inception ResNet v1 Architecture [4]
- **Training**: CASIA-WebFace [5], **LFW accuracy**: 0.987
- **Training**: MS-Celeb-1M [1], **LFW accuracy**: 0.992

Multitask Cascaded CNN (MTCNN) for face box detection

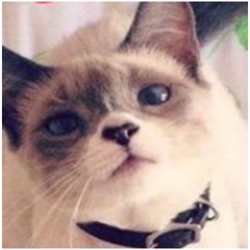
- Create a new dataset of all the pictures **containing only one face**
- Crop or enlarge these face to 182x182 pixel images using MTCNN
- Some false positive and false negatives, overall good
- Fortunately 95.1 % of the profiles I reviewed had at least a single picture with just one face
- Dataset is now 24,486 RGB (182x182x3) images of faces
- 8,130 profiles

Noise within Tinder One Face dataset

Here are a some of the MTCNN false positives.





Receiver operating characteristic (ROC)

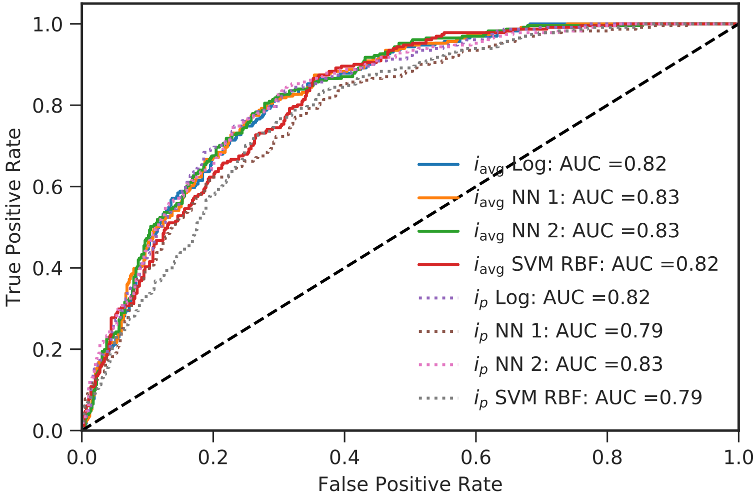
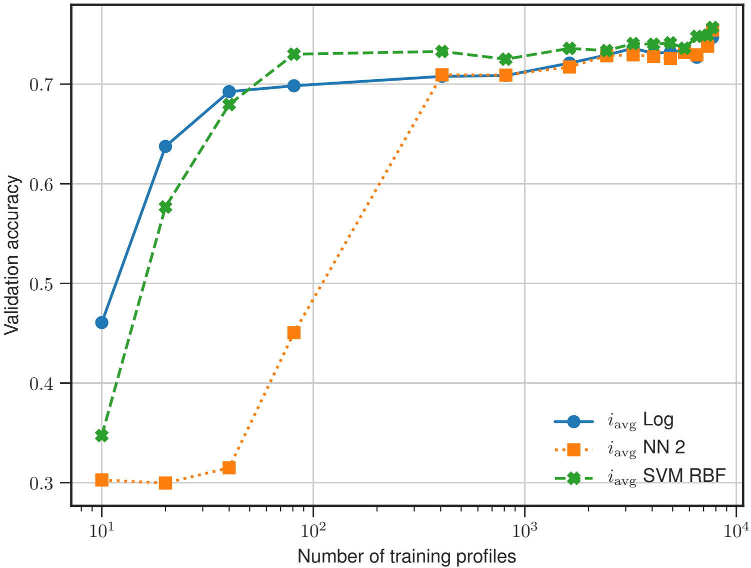


Figure 4: ROC and area under curve (AUC) for various classification models using a 10:1 train:test split.

Validation accuracy



What does this mean?

- **Finally managed to find a pattern**!
- 70% accuracy on just 80 profiles with logistic regression model
- Establish a point of diminishing marginal returns
- Class imbalances and weights to *Likes* and *Dislikes*

This is significantly better than randomly liking

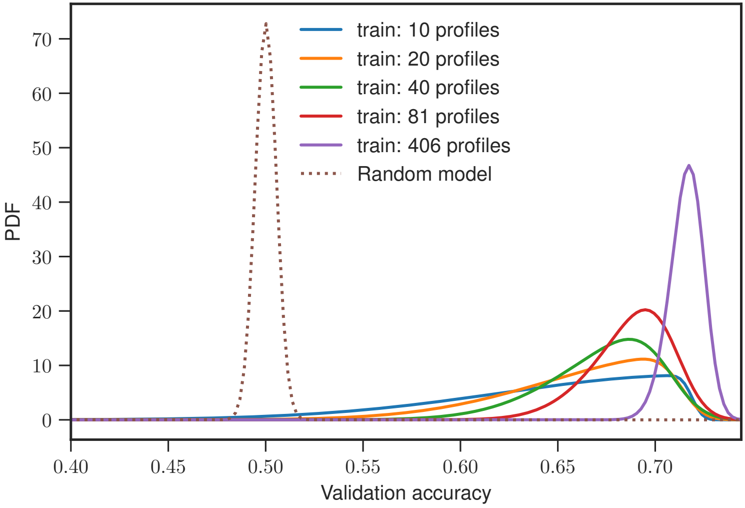


Figure 6: Probability density functions (PDF) for validation accuracy of classifiers trained on either 10, 20, 40, 81, and 406 profiles.

Build personalized machine learning models for Tinder based on your historical preference using Python.

1. A function to build a database which records everything about the profiles you've liked and disliked.
2. A function to train a model to your database.
3. A function to use the trained model to automatically like and dislike new profiles.

`https://github.com/cjekel/tindetheus`

tindetheus - LIVE DEMO

tindetheus browse

tindetheus train

tindetheus like

Future work

Reach out to me at cjekel@ufl.edu if you'd like to be involved

- Build pre-trained tindetheus models based on hot-or-not
- Consider more than just faces in the profiles
- NLP possibilities to include bio information

Everything here was done in Python

- Fall 2017 I created and taught 1 credit Python course
- Introduced the basic Python syntax
- Covered: matrix operations, plotting, statistics, optimization, scikit-learn, & more
- Material available online `https://jekel.me`

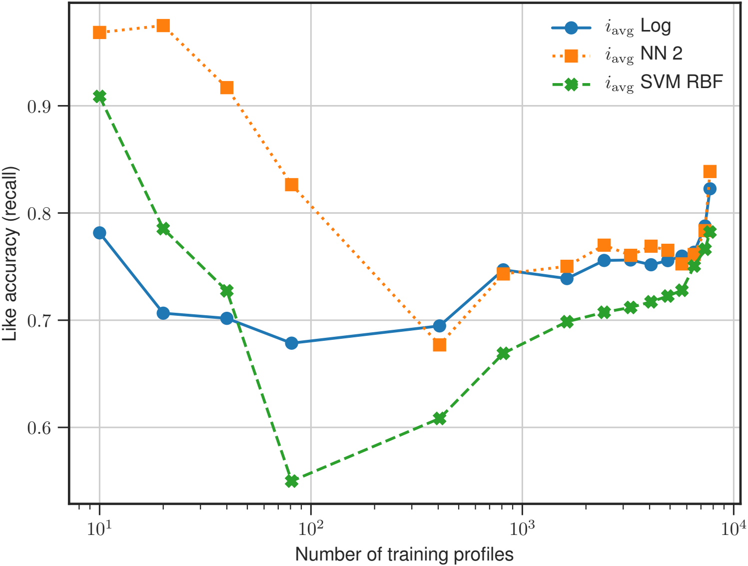
Closing details

I created a Python applications for users to build their own personalized models to Automatically like users on Tinder.

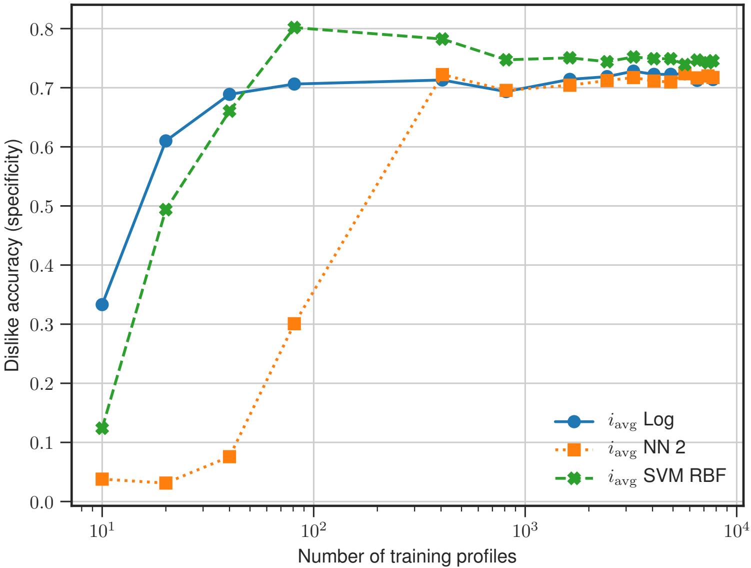
- FaceNet facial classification techniques
- 73% accuracy with training on just 80 profiles
- Various commercial applications
- My paper on this method `https://arxiv.org/abs/1803.04347`

Bonus slides...

Like accuracy (True positive rate)



Dislike accuracy (True negative rate)



Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao.
M{S}-{C}eleb-1{M}: A Dataset and Benchmark for Large Scale Face Recognition.
In *European Conference on Computer Vision*. Springer, 2016.

G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller.
Labeled faces in the wild: A database for studying face recognition in unconstrained environments.
Technical Report 07-49, University of Massachusetts, Amherst, oct 2007.

F. Schroff, D. Kalenichenko, and J. Philbin.
FaceNet: A Unified Embedding for Face Recognition and Clustering.
In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, jun 2015.

References (are in numerical order) ii

C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi.
Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.
In S. P. Singh and S. Markovitch, editors, *Proceedings of the Thirty-First {AAAI} Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, {USA.}*, pages 4278–4284. {AAAI} Press, 2016.

D. Yi, Z. Lei, S. Liao, and S. Li.
Learning Face Representation from Scratch.
2014.

K. Zhang, Z. Zhang, Z. Li, and Y. Qiao.
Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks.
IEEE Signal Processing Letters, 23(10):1499–1503, oct 2016.