Tecnológico de Monterrey

Instituto Tecnológico de Estudios Superiores de Monterrey
Campus Toluca

Desarrollo de Aplicaciones Avanzadas

Reconocimiento facial

Elaborado por:

Jose Israel Quintero Alfaro —A01366686
Bruno Cruz Mendoza —A01367102
Oscar Emilio Reyes Taboada —A01369421
Edgar Daniel Amaya —A01769048
Divad Alejandro Shriqui Garron —A01366907

Profesores

Marcial Roberto Leyva Fernández

4 de mayo de 2023

1. Introducción

El reconocimiento facial es una tecnología que cada día tiene más aplicaciones y más investigación, siendo uno de los principales enfoques de materias como la visión computacional o los modelos convolutivos de machine learning. Con aplicaciones como seguridad, autenticación de doble factor o reconocimiento, las posibilidades de estas tecnologías son cada vez mayores.

En este proyecto desarrollamos una aplicación web para la detección de un rostro en una fotografía tomada desde la misma aplicación, la cual tiene acceso a una base de datos con las matrículas y fotos de los alumnos del curso.

2. Desarrollo de la web app para el reconocimiento facial

Se decidió usar ciertas tecnologías para el reto las cuales son:

- Django
- HTML
- JavaScript

Aunque las últimas dos son prácticamente un sello del desarrollo de aplicaciones web, Django es un framework que no se usa tan comúnmente, pero en este caso, la naturaleza de nuestra aplicación prácticamente pedía el uso de Python, gracias a su versatilidad y los frameworks existentes para el uso de visión computacional y machine learning.

Para el desarrollo de esta aplicación tomamos una ruta muy convencional. Empezando por el front-end, y terminando con el desarrollo de back-end y la corrección de algunos errores causados por el mismo en las vistas.

2.1. Marco Teorico

Al empezar el desarrollo, realmente teniamos muy poco conocimento practico sobre la aplicacion de las tecnologias que utilizamos, aunque teniamos el conocimiento teorico. Utilizando modelos convolutivos como pudimos investigar en nuestro estado del arte, encontramos la base necesaria para hacer la aplicacion.

2.1.1. Machine Learning

El aprendizaje automático es una rama de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos que permiten a los ordenadores aprender a partir de datos. Estos modelos se construyen a partir de ejemplos de entrenamiento y se utilizan para hacer predicciones o tomar decisiones en situaciones similares a las que se utilizaron para su entrenamiento. Las aplicaciones del aprendizaje automático son variadas y van desde el reconocimiento de patrones en imágenes y audio hasta la optimización de procesos empresariales. Uno de los modelos más comunes es la red neuronal artificial, que se basa en el funcionamiento del cerebro humano y es capaz de aprender a partir de datos no estructurados. [Alpaydin, 2020]

2.1.2. Modelo Convolutivo

Los modelos convolutivos (CNNs) son un tipo de modelo de aprendizaje profundo que se utilizan comúnmente en tareas de procesamiento de imágenes y visión por computadora. Estos modelos se inspiran en el procesamiento visual del cerebro humano y utilizan capas convolutivas para extraer características de las imágenes y aprender representaciones cada vez más complejas a medida que se profundiza en la red.

La arquitectura de un modelo convolutivo típico consta de múltiples capas convolutivas seguidas de capas de submuestreo, capas de normalización y capas totalmente conectadas para la clasificación

final. Los modelos convolutivos han demostrado un rendimiento sobresaliente en una amplia gama de tareas de clasificación de imágenes, reconocimiento de objetos, detección de objetos y segmentación de imágenes.[LeCun et al., 1998]

2.1.3. Aplicacion Web

Las aplicaciones web son cada vez más populares debido a su capacidad para ofrecer servicios en línea y acceder a ellos desde cualquier lugar y dispositivo con conexión a internet. El desarrollo de aplicaciones web puede ser muy variado, desde aplicaciones simples basadas en HTML y JavaScript hasta aplicaciones más complejas con funcionalidades avanzadas basadas en frameworks y tecnologías modernas. Las aplicaciones web también se benefician de las ventajas de la nube, como la escalabilidad y el bajo costo de infraestructura.

Un ejemplo de aplicación web popular es Google Docs, que permite a los usuarios crear, editar y colaborar en documentos en línea en tiempo real. Otras aplicaciones web populares incluyen plataformas de comercio electrónico como Amazon y eBay, y aplicaciones de productividad como Trello y Asana. [Rajput and Singh, 2021]

2.1.4. Python

Python es un lenguaje de programación de alto nivel utilizado en diversas aplicaciones, desde el desarrollo web hasta el análisis de datos y la inteligencia artificial. Su popularidad se debe en gran medida a su facilidad de uso y legibilidad, lo que lo convierte en una excelente opción para principiantes y expertos por igual. Además, cuenta con una gran cantidad de bibliotecas y frameworks para ayudar en el desarrollo de aplicaciones, como Flask para aplicaciones web y TensorFlow para machine learning. [Van Rossum and Matthes, 2019]

2.1.5. Django

Django es un popular framework web de alto nivel escrito en Python que sigue el patrón de diseño MVT (Model-View-Template). Este framework se utiliza comúnmente para desarrollar aplicaciones web complejas y escalables de manera rápida y eficiente. Django proporciona una amplia variedad de características útiles para el desarrollo web, como un ORM (Object-Relational Mapping) integrado, un sistema de autenticación de usuarios, una capa de abstracción de base de datos y un sistema de enrutamiento. [Holovaty and Kaplan-Moss, 2004]

2.2. Diagramas

2.2.1. Diagrama

(Ver la Figura 1)

Como podemos ver en la Figura 1, el diagrama de flujo comienza con el usuario ingresando a la aplicación y luego introduciendo una foto para analizar, así como una placa para comparar. Después, el programa realiza todo el análisis necesario para finalmente mostrar los resultados en una página separada en el frontend.

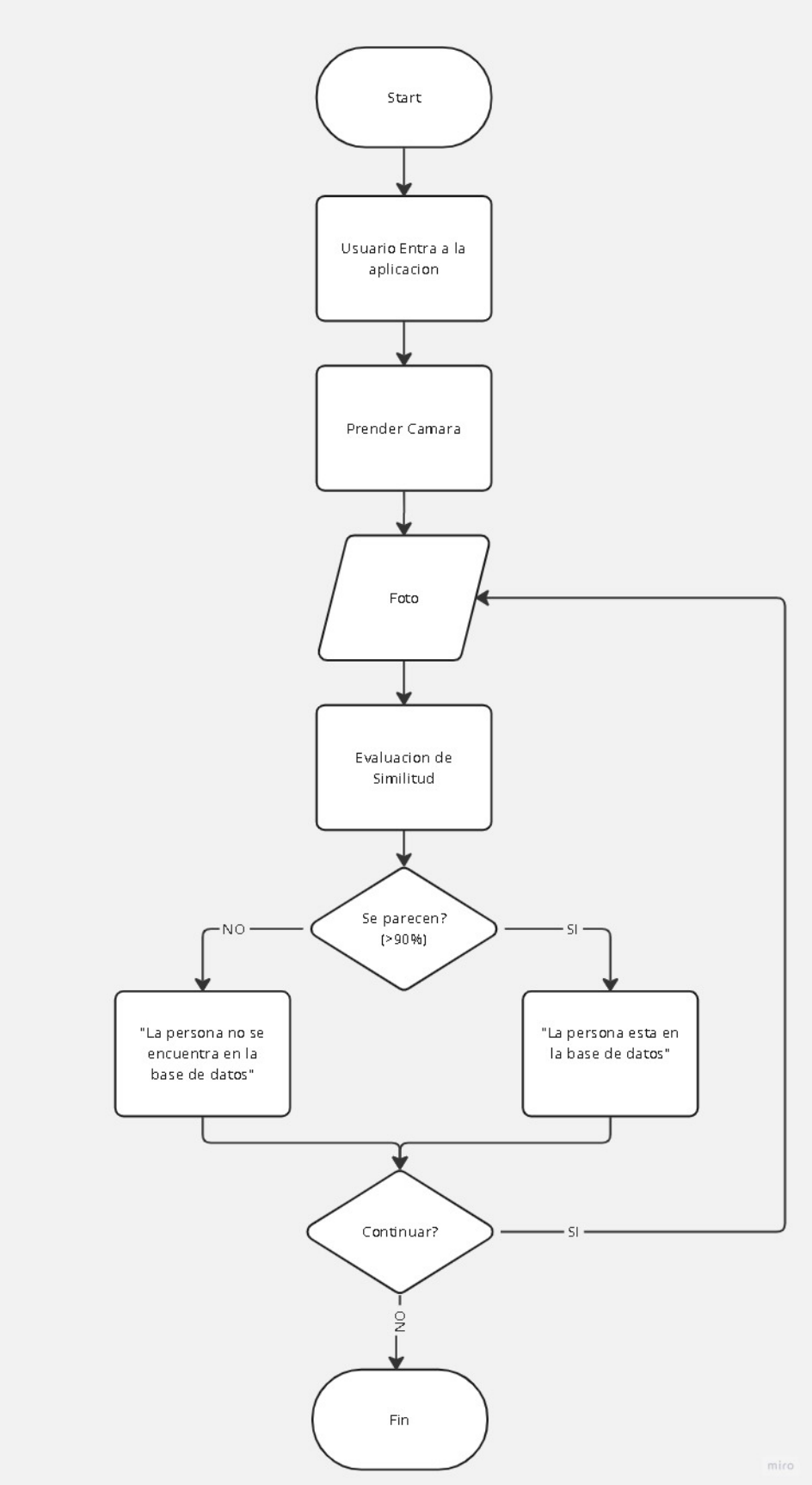

Figura 1: Este es el diagrama de flujo de la aplicacion

2.3. Funcionamiento, Modelo y Justificación

Debido a las limitaciones de tiempo del proyecto, optamos por un desarrollo que únicamente funciona de manera local, debido a que el proceso de alojar la página para su funcionamiento en la nube hubiera complicado de manera exponencial la implementación del proyecto. Es por esto que solo funciona dentro del servidor "localhost", pero importando el proyecto su funcionamiento es correcto.

El frontend funciona desde Django, gracias a la naturaleza MVT del framework. Este contiene codificación en HTML, CSS y JS; todo el backend funciona dentro de Python. El proceso comienza desde la página, donde se puede dar clic en ïniciar cámara"para poder tomar la foto desde la webcam del dispositivo, esto hace que el funcionamiento sea muy intuitivo. También se puede apagar la cámara para la comodidad del usuario. (figura 2)

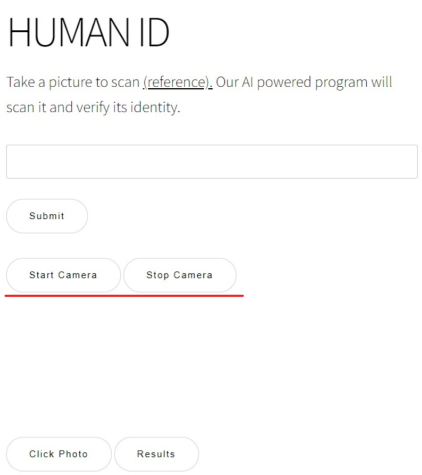

Figura 2: Podemos ver remarcados los botones que inician y desactivan la camara

Los resultados son mostrados en una página aparte para poder enseñar más datos relevantes a la investigación. (figura 3)

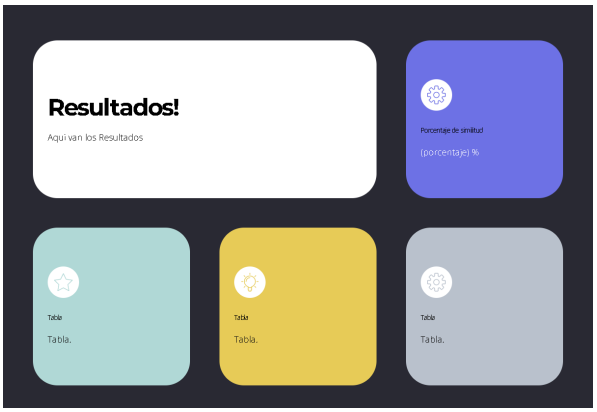

Figura 3: View con los resultados

En backend se corre, de manera paralela al frontend, los scripts de Python que hacen las comparaciones, usando un modelo convolucional para generar el cálculo de probabilidad de que la persona que aparece en la foto esté en la base de datos. (figura 4) El programa compara la foto con la base de datos que contiene las imágenes de las personas que alimentan el modelo, estudiantes y profesor.

```python
# Load the new image and convert it to a numpy array
new_img = cv2.imread("testpics/JIQA.jpg")
new_img = cv2.cvtColor(new_img, cv2.COLOR_BGR2RGB)  # convert to RGB
size = (128, 128)
new_img = cv2.resize(new_img, size)  # resize to match the size of images in the dataset
new_img_array = numpy.asarray(new_img)

print(new_img.size)

# Transform the new image array using the PCA model
new_img_transformed = Model.transform(new_img_array.reshape(1, -1))

# # Transform the images of the specified ID using the PCA model
specified_X = numpy.asarray(filtered_df.iloc[:,1:])
specified_X_transformed = Model.transform(specified_X)

Sim = []
for xi in specified_X_transformed:
  sim = Similiarity(new_img_transformed, xi)
  Sim.append(sim)

# # Get the indices of the most similar images
Idx = numpy.argsort(Sim)

# # Print the top 5 most similar images of the specified ID
print(filtered_df.iloc[Idx[0:5]])

```

Figura 4: Código de python que procesa las imágenes para su análisis.

2.4. Resultados

En un principio, la implementación fue muy complicada debido a algunas deficiencias que teníamos tanto en el diseño como en la implementación inicial, pero después de arreglar estos contratiempos, el desarrollo empezó a verse cada vez más fluido.

Una de estas deficiencias es que no teníamos muy claro el funcionamiento del resultado de la distancia euclidiana, por lo que el porcentaje que nos daba de similitud era o muy alto o muy bajo. Sin embargo, después de un análisis matemático del modelo, logramos aproximar los resultados a cifras más cercanas a la realidad.

Probamos con muchos casos distintos, con nuestros rostros y con rostros generados por inteligencias artificiales, y logramos resultados satisfactorios en el aspecto de que detecta cuando la foto proporcionada es idéntica o muy similar al modelo de aprendizaje (figura 5). Si el rostro no forma parte de la muestra o no se parece a ninguno, el resultado es la correcta identificación de la falta de similitud con el modelo (figura 6).

Figura 5: Prueba exitosa

Figura 6: Prueba exitosa (de no similitud)

En conclusión, podemos decretar que, a pesar de las dificultades de implementación, los resultados del proyecto son correctos y eficaces en la detección de similitud en rostros.

2.5. Galeria

Galeria de fotos de la aplicación

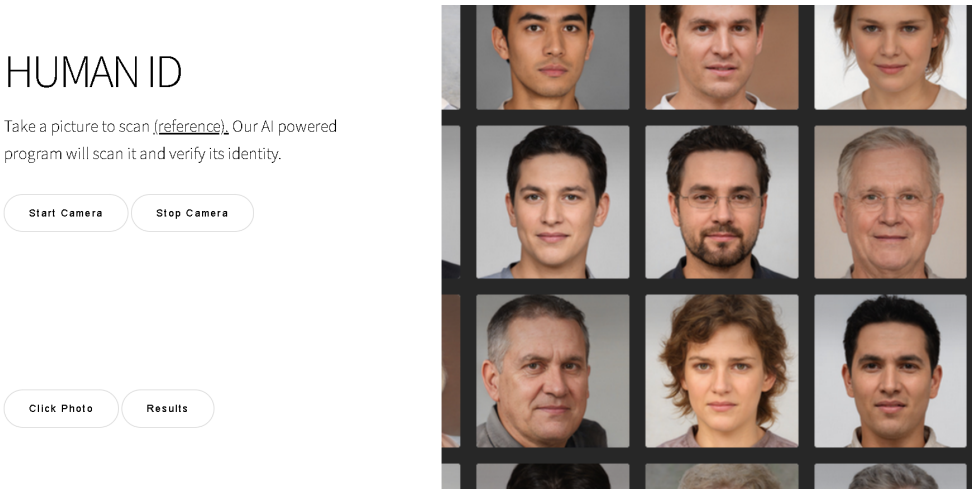

Figura 7: Pagina del app completa

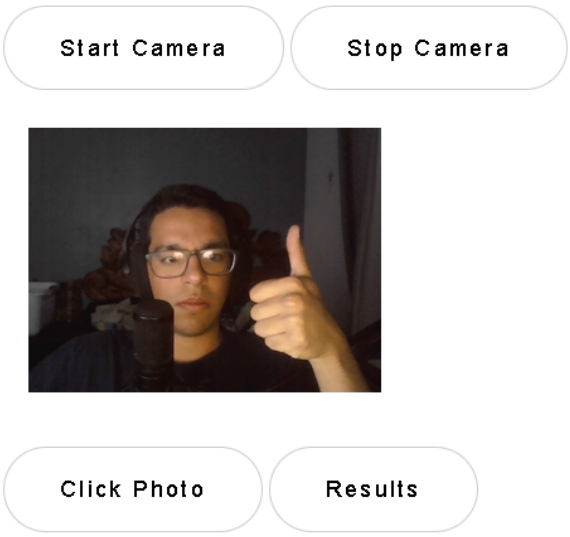

Figura 8: Cámara funcionando

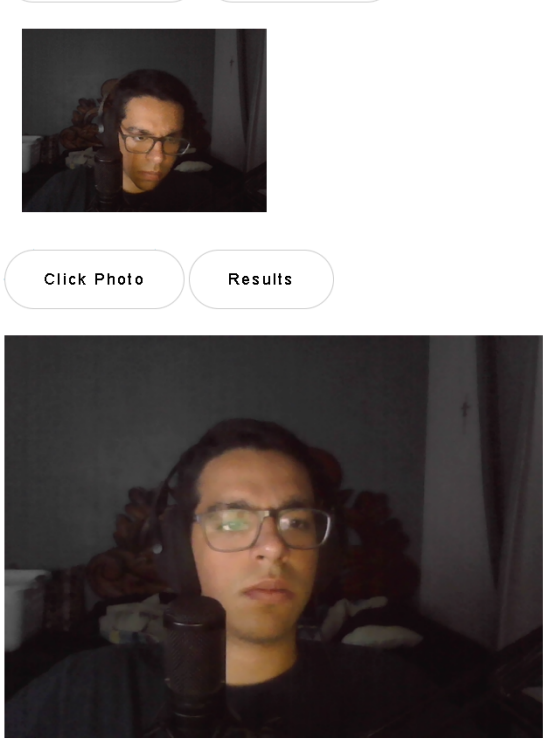

Figura 9: Cámara funcoinando y se tomó una foto

3. Link del Github Pages y el Github con las instrucciones

- Repositorio

- Página con presentación

4. Reflexión

4.1. Reflexión Oscar:

El uso de este tipo de tecnologías tiene cada día más aplicaciones, y el uso de los modelos correctos puede significar toda la diferencia entre una situación gratificante o pobre.

Al usar estos frameworks y modelos para el reconocimiento facial nos damos cuenta de que esto bien puede ser el futuro más cercano de la visión computacional, no tanto en el reconocimiento de identidad, sino en el reconocimiento de facciones, o incluso emociones. Esto se puede usar en todo tipo de campos, desde la mercadotecnia al cuantificar las emociones de alguien al ver un anuncio, hasta la posibilidad de utilizar este tipo de tecnologías para reconocer delincuentes o personas que estén en una base de datos de esta naturaleza. Este proyecto es solo la punta del iceberg de este gigante de la computación que es el machine learning.

4.2. Reflexión Bruno:

Creando esta aplicación de reconocimiento facial con Django y diversas bibliotecas de Python, aprendí mucho sobre el mundo del aprendizaje automático y la inteligencia artificial. Fue una excelente manera de poner en práctica mis habilidades de programación y aprender nuevas técnicas y bibliotecas. Sin embargo, debo admitir que el uso de Django fue un poco difícil al principio. Hubo algunos desafíos técnicos que tuve que superar antes de poder construir la aplicación. Aun así, todo el proceso fue muy divertido y pude crear una aplicación funcional y atractiva. En general, estoy muy contento con el resultado final de esta aplicación. Además, ahora tengo una mejor comprensión de cómo funciona el reconocimiento facial y cómo se puede implementar en aplicaciones del mundo real. Definitivamente recomendaría a cualquier persona interesada en el aprendizaje automático que pruebe este proyecto como una forma divertida y práctica de aprender.

4.3. Reflexión Divad:

La inteligencia artificial cada vez una parte más integra de nuestro día a día, la implementación de esta tecnología es cada vez más útil en la vida cotidiana. El uso de esta es primordial para la seguridad de las personas y sus dispositivos, aunque ahora la precisión puede no ser tan exacta, en un futuro muy cercano con la cantidad de información que se le esta brindando a los modelos de esta tecnología, esta puede llegar a ser mas exacta que el mismo ojo humano. Durante este proyecto he estado mas atento a la hora de el uso de las librerías y documentar estas para no tener problemas de compatibilidad mas adelante. Al hacer uso de nuestro sistema para reconocimiento facial nos pudimos dar cuenta que existen ciertas limitaciones en el hardware que podrían ocasionar problemas a la hora de analizar, por ejemplo la resolución de la cámara de mi laptop era tan baja que al ingresar la imagen en el sistema era difícil pero no imposible que me reconozca los rasgos faciales.

4.4. Reflexión Israel:

Experimentar con nuevas tecnologías siempre es gratificante e impresionante, pero también nos permite visualizar la facilidad con la cual se pueden implementar estas herramientas y su fácil acceso para el resto del mundo. Esto nos hace reflexionar sobre el posible uso maligno que puede conllevar

el uso de estas tecnologías. Con un entrenamiento limitado y pocas fotos, logramos obtener una similitud y reconocimiento sorprendentemente alto. No podemos evitar asombrarnos de la velocidad a la cual avanza la tecnología. Esto también nos hace reflexionar sobre el cuidado que debemos tener al implementar nuevas tecnologías y nuestra responsabilidad al hacerlo. Es importante tener en cuenta la privacidad y los derechos humanos en el diseño y uso de estas herramientas, y trabajar para mitigar cualquier posible impacto negativo. La tecnología puede ser transformadora, pero debemos utilizarla de manera responsable y ética para el bien común.

4.5. Reflexión Daniel:

Usar tecnología de reconocimiento facial, dentro de un entorno de DJANGO, resulto en una tarea más extensa que usar otros medios, especialmente en la parte de backend, más sin embargo se observó un procesamiento de datos mucho más rápido, teniendo esto en cuenta, en futuros proyectos, considerar trabajar en un ambiente computacional como la nube, haría que el rendimiento del proyecto se vea optimizado con el uso de menor recursos posibles.

Referencias

[Alpaydin, 2020] Alpaydin, E. (2020). *Introduction to Machine Learning*. MIT press.

[Holovaty and Kaplan-Moss, 2004] Holovaty, A. and Kaplan-Moss, J. (2004). Django: Web framework for perfectionists with deadlines. *ACM SIGWEB Newsletter*, 13(1):5–5.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[Rajput and Singh, 2021] Rajput, R. D. and Singh, N. K. (2021). Web development: Tools, techniques and applications. *International Journal of Emerging Trends Technology in Computer Science*, 10(5).

[Van Rossum and Matthes, 2019] Van Rossum, E. and Matthes, E. (2019). *Python Crash Course: A Hands-On, Project-Based Introduction to Programming*. No Starch Press.