
Desarrollo de web scraping sobre papers

Departamento de Ingeniería de la Información y las Comunicaciones

Universidad de Murcia – Grado en Ingeniería Informática

Paduraru, Cosmin Marian –cosminmarian.paduraru@um.es

Bajo la tutela de: Juan Antonio Botía Blaya

Contenido

| | |
|--------------------------------------|---|
| 1. Web scraping PapersWithCode | 3 |
| 1.1 Explicación general..... | 3 |
| 1.2 Manual de usuario..... | 6 |
| 2. Bibliografía | 8 |

1. Web scraping PapersWithCode

1.1 Explicación general

Web scraping es la técnica dentro del mundo de la programación que se encarga de extraer información de distintos sitios web. En mi caso usaré esta técnica para la extracción de información de distintos papers que están reunidos y recopilados dentro del portal de: <https://paperswithcode.com/>

De esta forma podré extraer información relevante sobre un paper determinado, la información que extraeré será la siguiente:

- Título
- Resumen
- Enlace del Artículo
- Enlace del PDF asociado
- Enlace del Código asociado
- Autores
- Fecha de publicación
- Métodos
 - Nombre del método
 - Enlace donde se explica el método
 - Definición del método

Como se puede observar se extraen varios datos dado un enlace determinado. Un ejemplo de un paper en la página web es el siguiente (*Figura 2*). Como se puede observar la información está esparcida por toda la página web. Lo que hace mi programa es recopilar toda la información que nos interesa y extraerla en un documento Excel.

| | A | B | C | D | E | F | G |
|---|----------|-----------|---|---|---|------------|-------------|
| 1 | TITULO | ABSTRACT | PAPER_LINK | PDF_LINK | CODE_LINK | AUTHORS | FECHA_PUB |
| 2 | VideoGPT | We presen | https://paper | https://ar | https://github | ["Wilson Y | 20 Apr 2021 |
| 3 | | | | | | | |
| 4 | | Nombre | Link | Definicion | | | |
| 5 | MET 1 | Axial | https://paper | Axial attention is a simple generalization of s | | | |
| 6 | MET 2 | VQ-VAE | https://paper | VQ-VAE is a type of variational autoencoder t | | | |

Figura 1

Cuya estructura es la siguiente (*Figura 1*). Como se puede ver en la fila numero dos, se tiene toda la información que hace referencia al propio paper (*nombre, resumen, enlaces...*). Por otro lado, a partir de la la fila cuatro, se almacena toda la información respectiva a los métodos que se han utilizado en el paper y que se mencionan con anterioridad.

Para este programa he decido utilizar el lenguaje de programación Python, que es muy versátil y de alto nivel, también podría haberlo hecho con cualquier otro lenguaje, pero es el que mejor se adecua a la situación y el que actualmente mejor manejo, por lo que me parecía la opción más adecuada.

VideoGPT: Video Generation using VQ-VAE and Transformers

20 Apr 2021 · Wilson Yan, Yunzhi Zhang, Pieter Abbeel, Aravind Srinivas · [Edit social preview](#)

We present VideoGPT: a conceptually simple architecture for scaling likelihood based generative modeling to natural videos. VideoGPT uses VQ-VAE that learns downsampled discrete latent representations of a raw video by employing 3D convolutions and axial self-attention... [read more](#)



Code

[Edit](#)



★ 309

PyTorch

Tasks

[Edit](#)



Datasets

[Edit](#)



Results from the Paper

[Edit](#)

Ranked #2 on Video Generation on BAIR Robot Pushing

[Get a GitHub badge](#)

| Task | Dataset | Model | Metric Name | Metric Value | Global Rank | Benchmark |
|------------------|---|----------|-----------------|--------------|-------------|-------------------------|
| Video Generation | BAIR Robot Pushing | VideoGPT | FVD score | 103.3 | # 2 | Compare |
| Video Generation | UCF-101 16 frames, 128x128, Unconditional | VideoGPT | Inception Score | 24.69 | # 3 | Compare |

Methods

[Edit](#)

Axial • VQ-VAE

Figura 2

Junto con Python, he utilizado las siguientes librerías:

- requests
- BeautifulSoup
- regex
- xlswriter
- sys
- getopt
- os

Procedo a explicar brevemente cada una de las librerías utilizadas, aunque no me voy a explayar demasiado en todas, puesto que la documentación de estas está en internet disponible.

- *requests*: Sirve principalmente para hacer solicitudes a una *url* y mediante la cual podemos obtener el código http asociado con dicha solicitud y mediante el cual extraeremos la información que nos interesa.
- *beautifulSoup*: En mi caso la utilizaré para filtrar el contenido que nos interesa de dicha solicitud, mediante las funciones que tiene asociadas, principalmente *find* y *find_all* en mi caso.
- *regex*: Usada para detectar patrones en cadenas mediante el uso de expresiones regulares.
- *xlsxwriter*: Es la librería encargada de exportar la información deseada a un fichero Excel.
- *sys*, *getopt* y *os*: Serán las librerías encargadas de hacer la funcionalidad de encontrar varios papers mediante una palabra o grupo de palabras determinado. Básicamente *os* será utilizada para llamar al sistema mientras que *getopt* será utilizada para gestionar las llamadas por parámetro. Por otro lado, *sys* será encargado en mi caso de recibir los argumentos.

Hay que destacar que hay dos programas, (*paper_ws.py*) uno que se utilizará para extraer la información relevante de un solo paper, mientras que habrá otro (*multiple_papers.py*) encargado de extraer la información relevante de varios papers. Este último hará uso del primero, por lo que para que funcione tendrán que estar ambos dentro del mismo directorio.

La estructura que sigue el código en general es bastante intuitiva. Hay varias funciones que se encargan de extraer la información del código html, estas funciones empiezan por *get...* En la siguiente sección de código podemos ver un ejemplo de ello. En este se obtiene la definición de un método que se esté utilizando en el paper.

```
def getDefinicionMetodo(link):

    lAux = body(link).find('div', class_ = 'method-content').find('div', class_ = 'row').find_all('p')
    resultado = ''
    for element in lAux:
        resultado = resultado + element.text + '\n'
    return resultado
```

Mientras que por otro lado está la función encargada de exportar la información extraída por los métodos anteriores en un fichero Excel. La función no es muy complicada, pero se divide en dos partes, la escritura de los títulos de lo que se está escribiendo y la escritura de la información. Lo primero se hace de la siguiente forma:

```
worksheet.write('D1', 'PDF_LINK', bold)
```

Mientras que lo segundo se hace de la siguiente:

```
for nombre in metodos:
    worksheet.write(fila, columna, 'MET ' + str(fila-3), bold)
    worksheet.write(fila, columna + 1, nombre)
    worksheet.write(fila, columna + 2, metodos[nombre][0])
    worksheet.write(fila, columna + 3, metodos[nombre][1])
    fila += 1
```

Se puede observar que para gestionar la información de los métodos simplemente se va escribiendo en una nueva fila su información, hasta que no queden más métodos que escribir.

1.2 Manual de usuario

- *paper_ws.py*

Este programa será el encargado de extraer la información sobre un paper determinado sobre el cual se proporcione un enlace, en caso de que no se proporcione ningún parámetro se ejecutará uno de ejemplo sobre la siguiente url: <https://paperswithcode.com/paper/video-gpt-video-generation-using-vq-vae-and-transformers>. Mediante el cual podremos ver su funcionamiento. Una vez se ejecute el programa (2-5 seg dependiendo del ordenador y la conexión a internet) saldrá un mensaje por pantalla indicando que se ha generado el fichero de salida y sobre que url, en nuestro caso sería el siguiente mensaje:

Fichero output.xlsx creado con éxito del paper: VideoGPT: Video Generation using VQ-VAE and Transformers

Además, cuando abramos el fichero *output.xlsx*, podremos ver su contenido (*Figura 1*). Ahora supongamos que vamos a utilizarlo pasándole parámetros, los que soporta actualmente son los siguientes:

- *-h* Que nos proporciona la información de cómo se utiliza el script.
- *-o* nombre del fichero de salida (no hace falta incluir *.xlsx* solo el nombre, es decir: *fichero.xlsx*, solo la parte subrayada)
- *-l* El enlace sobre el que se desea hacer funcionar el programa.

Procedo a mostrar un ejemplo:

```
python3 paper_ws.py -o salida_ejemplo -l https://paperswithcode.com/paper/emerging-properties-in-self-supervised-vision
```

Cuya salida por pantalla será la siguiente:

Fichero salida_ejemplo.xlsx creado con éxito del paper: Emerging Properties in Self-Supervised Vision Transformer

Y el contenido del fichero se muestra a continuación:

| TITULO | ABSTRACT | PAPER_LINK | PDF_LINK | CODE_LINK | AUTHORS | FECHA_PUB | | | |
|----------|------------------------------|---|---|---|----------------|-------------|--|--|--|
| Emerging | In this paper | https://paperswithcode.com/paper/emerging-properties-in-self-supervised-vision | https://arxiv.org/abs/2104.02111 | https://github.com/mathilde294/vision-transformer | Mathilde Caron | 29 Apr 2021 | | | |
| | Nombre | Link | Definición | | | | | | |
| MET 1 | Adam | https://paperswithcode.com/paper/adam | Adam is an adaptive learning rate optimization algorithm that utilises both moment estimates and variance estimates. | | | | | | |
| MET 2 | BPE | https://paperswithcode.com/paper/byte-pair-encoding | Byte Pair Encoding, or BPE, is a subword segmentation algorithm that encodes a sequence of characters into a sequence of subwords. | | | | | | |
| MET 3 | Dense Connections | https://paperswithcode.com/paper/dense-connections | Dense Connections, or Fully Connected Connections, are a type of layer in a deep neural network. | | | | | | |
| MET 4 | Dropout | https://paperswithcode.com/paper/dropout | Dropout is a regularization technique for neural networks that drops a unit (along with its connections) during training. | | | | | | |
| MET 5 | k-NN | https://paperswithcode.com/paper/k-nearest-neighbors | k-Nearest Neighbors is a clustering-based algorithm for classification and regression. | | | | | | |
| MET 6 | Label Smoothing | https://paperswithcode.com/paper/label-smoothing | Label Smoothing is a regularization technique that introduces noise for the labels during training. | | | | | | |
| MET 7 | Layer Normalization | https://paperswithcode.com/paper/layer-normalization | Unlike batch normalization, Layer Normalization directly estimates the normalization statistics for each layer. | | | | | | |
| MET 8 | Multi-Head Attention | https://paperswithcode.com/paper/multi-head-attention | Multi-head Attention is a module for attention mechanisms which runs through multiple attention heads. | | | | | | |
| MET 9 | Residual Connections | https://paperswithcode.com/paper/residual-connections | Residual Connections are a type of skip-connection that learn residual functions with respect to the layer inputs. | | | | | | |
| MET 10 | Scaled Dot-Product Attention | https://paperswithcode.com/paper/scaled-dot-product-attention | Scaled dot-product attention is an attention mechanism where the dot product of the query and key is scaled by the square root of the dimensionality of the keys. | | | | | | |
| MET 11 | Softmax | https://paperswithcode.com/paper/softmax | The Softmax output function transforms a previous layer's output into a vector of probabilities. | | | | | | |
| MET 12 | Transformer | https://paperswithcode.com/paper/transformer | A Transformer is a model architecture that eschews recurrence and instead relies on attention mechanisms. | | | | | | |
| MET 13 | Vision Transformer | https://paperswithcode.com/paper/vision-transformer | The Vision Transformer is a model for image classification that employs a Transformer architecture. | | | | | | |

Figura 3

Como se puede ver, este paper hace uso de muchísimos más métodos y por lo tanto al ejecutarlo es algo más lento, ya que tiene que acceder a cada método y extraer información de este.

- *multiple_papers.py*

Este programa servirá para extraer la información de varios papers que incluyan alguna palabra, es decir es un programa que nos permite filtrar papers gracias a las palabras. Para esto he utilizado el programa anterior y otro paper.

Cuando ejecutemos el programa se nos preguntará sobre que palabras queremos hacer la búsqueda:

Introduzca la palabra/frase por la que desea buscar papers:

Introduciremos el conjunto de palabras sobre el que deseamos realizar la búsqueda y tendremos una salida similar a esta:

```
Introduzca la palabras/frase por la que desea buscar papers:
rgb clear
Buscando y haciendo metodo de busqueda...
Fichero 1.xlsx creado con exito del paper: DensePose: Dense Human Pose Estimation In The Wild
Fichero 2.xlsx creado con exito del paper: KeyPose: Multi-View 3D Labeling and Keypoint Estimation for Transparent Objects
Fichero 3.xlsx creado con exito del paper: First-Person Hand Action Benchmark with RGB-D Videos and 3D Hand Pose Annotations
Fichero 4.xlsx creado con exito del paper: Self-supervised Co-training for Video Representation Learning
Fichero 5.xlsx creado con exito del paper: Efficient Plane-Based Optimization of Geometry and Texture for Indoor RGB-D Reconstruction
Fichero 6.xlsx creado con exito del paper: Deep Multimodal Fusion by Channel Exchanging
Fichero 7.xlsx creado con exito del paper: Relative Camera Pose Estimation Using Convolutional Neural Networks
Fichero 8.xlsx creado con exito del paper: Deep Interactive Object Selection
Fichero 9.xlsx creado con exito del paper: Confidence Propagation through CNNs for Guided Sparse Depth Regression
Fichero 10.xlsx creado con exito del paper: Chained Multi-stream Networks Exploiting Pose, Motion, and Appearance for Action Classification and Detection
```

Por cada resultado de la búsqueda saldrá un fichero *n.xlsx* y por pantalla se nos dirá la información de que paper se realizado la búsqueda y se ha extraído la información.

2. Bibliografía

<https://paperswithcode.com/>

<https://realpython.com/beautiful-soup-web-scraper-python/>

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

<https://www.codegrepper.com/code-examples/python/bs4.element.tag+methods>

<https://xlsxwriter.readthedocs.io/tutorial02.html>

<https://pymotw.com/2/getopt/>

https://es.wikipedia.org/wiki/Web_scraping