

# **Narzędzie do generowania tytułów publikacji naukowych**

Patryk Kordek,  
Hubert Gawczyński

## Spis treści:

1. Abstrakt .....	
2. Wstęp .....	
2.1 Cel .....	
2.2 Zakres .....	
2.3 Metodyka .....	
3. Realizacja .....	
3.1 Pobieranie danych treningowych .....	
3.2 Uczenie modelu GPT-2 .....	
3.3 Generowanie tytułu na podstawie wytrenowanego modelu .....	
4. Podsumowanie .....	
5. Bibliografia .....	

## 1. Abstrakt

Przedstawiony projekt eksperymentuje z opracowaniem rozwiązania umożliwiającego generowanie nazw naukowych na podstawie najnowszych tendencji. Przewidywanie postępu w badaniach naukowych jest ważnym, ale trudnym celem. Projekt wykorzystuje dane z artykułów naukowych z pierwszej połowy 2019 roku do generowania sugerowanych tytułów prac z obszaru przetwarzania języka naturalnego. W ramach projektu użyto bazy danych ArXiv, modelu GPT-2 oraz języka Python.

## 2. Wstęp

### 2.1 Cel pracy

Celem naszej pracy jest stworzenie narzędzia do generowania tytułów publikacji naukowych.

### 2.2 Zakres pracy

Zakres obejmuje wytrenowanie i przetestowanie modelu gpt2 oraz analiza sensowności wygenerowanych tytułów.

### 2.3 Metodyka

Metodyka niniejszej pracy opiera się na przeprowadzeniu analizy danych, trenowaniu modelu oraz generowaniu tytułów, dbając o spójność i jakość wyników. W pracy użyliśmy danych z bazy danych ArXiv oraz modelu do trenowania GPT-2.

### 3. Realizacja

Realizacja pracy polega na: pobieraniu danych treningowych, uczeniu modelu oraz generowaniu tytułów na wytrenowanej sieci. Kod źródłowy bazuje na projekcie gpt-paper-title-generator.

#### 3.1. Pobranie Danych Treningowych

Dzięki bibliotece arxivscraper pozyskujemy dane z archiwum naukowych preprintów – ArXiv. Tytuły filtrujemy po kategorii i dacie publikacji. Na poniższym zrzucie ekranu widać dane pobrane w przedziale od 2019-05-01 do 2019-07-03 w kategorii fizyki kwantowej.

```
import arxivscraper as ax
import numpy as np

# scraper for arxiv physics
scraper = ax.Scraper(category='physics', date_from='2019-05-01',
                      date_until='2019-07-03', t=10,
                      filters={'categories':['quant-ph']})

output = scraper.scrape()

titles = [' '.join(o['title'].split()) for o in output]
np.savetxt('titles_ref.csv', np.array(titles), fmt='%s')

fetching up to 1000 records...
fetching up to 2000 records...
Got 503. Retrying after 10 seconds.
fetching up to 2000 records...
fetching up to 3000 records...
Got 503. Retrying after 10 seconds.
fetching up to 3000 records...
fetching up to 4000 records...
fetching up to 5000 records...
fetching up to 6000 records...
Got 503. Retrying after 10 seconds.
fetching up to 6000 records...
fetching up to 7000 records...
fetching up to 8000 records...
fetching up to 9000 records...
fetching up to 10000 records...
Got 503. Retrying after 10 seconds.
fetching up to 10000 records...
fetching up to 11000 records...
Got 503. Retrying after 10 seconds.
fetching up to 11000 records...
fetching up to 12000 records...
fetching up to 13000 records...
Got 503. Retrying after 10 seconds.
fetching up to 13000 records...
fetching is completed in 224.6 seconds.
Total number of records 1363
```

Rys.1 Pobieranie danych

### 3.2 Uczenie modelu GPT-2.

Do trenowania wykorzystaliśmy model sieci neuronowej GPT-2 posiadający 117 milionów parametrów. Następnie wykorzystaliśmy metodę `finetune()`, której parametrami są między innymi plik z pobranymi z ArXiv tytułami, liczba generacji czy częstotliwość zapisu modelu. Liczba generacji uczących została ustawiona na 1000, a co dziesiątą iterację aktualny model oraz wygenerowane tytuły publikacji są zapisywane do pliku.

```
import gpt_2_simple as gpt2
import tensorflow as tf

model_name = "117M"
gpt2.download_gpt2(model_name=model_name)

sess = gpt2.start_tf_sess()
gpt2.finetune(sess,
               'titles_ref.csv',
               model_name=model_name,
               steps=1000,
               save_every=10,
               sample_every=10,
               reuse=tf.compat.v1.AUTO_REUSE)

gpt2.generate(sess)
```

Rys.2 Pobieranie i uczenie modelu

Tytuły stworzone podczas procesu uczenia mogą być przedstawione po wcześniejszym usunięciu niepotrzebnych znaków co robimy w poniższym kodzie za pomocą pętli for.

```
sample_file = 'samples/run1/samples-51'
t = open(sample_file, 'r').read()

for s in ['endoftext', 'startoftext', '<|', '>']:
    t = t.replace(s, '')
for title in t.title().split('\n')[1:]:
    if not title == '':
        print('- ' + title)

- Math|Summoning For Finite Quantities Of Unruh Information Via An Arbitrary Localizer
- Efficiently Generating And Benchmarking Many-Body Processes With Quantum Spin Information
- Enhanced Control And Robustness Of Quantum Control Ensemble With Spin And Beam Fields
- Anomalous Noise Absorption In Fermions: From Non-Hermitian Quantum Mechanics To Interferometric Quantum Electrodynamics
- Experimental Detection Of Entanglement As The Key To Causality In A Non-Markovian World Of Three Dimensional Qubits
- A Single-Photon Optomechanics Treatment For The Hamiltonians
- Quantum And Bosonic Time Crystals
- A Quantum Nonlocality Unit And A Symmetries Rule
- Geometric Properties Of The Tensor Network In Finite Dimension Time Order
- Robust Quantum Key Distribution With Non-Commutativity
- Generalized Quantum Key Distribution With Coherence
- Quantum State Generation Via Optomechanics
- Quantum Correlations Of Nonreciprocal Channels By Using Multi-Level Quantum Key Distribution
- An Integrable Bohm-Barrier Quantum Algorithm For Computing  $\$P\$ States
- A Quantum Simulation Of The Eigenstate Distribution Of A Josephson Equation And A Waveguide Waveguide
- A Top-Symmetric Quantum One-Shot Engine With Near-Unitary Symmetry
- Quantum And Theorems Of Quantum Mechanics
- On The Atomic Level A Model Of The Causal Effects Of Action Points Is An Attractive Approach To The Problems Of Computation
- A Coherent Model Of The Entanglement-Induced Fermion Qubit System
- A Quantum-Mechanical Simulator Of Molecular Dynamics Using Optomechanics Of A Quantum Dot
- Geometric Scaling And Quantum Random Walk
- Quantum Computing Power (Billion-Barrier) With An Approach To Relativistic Quantum Dynamics
- Universal Hamiltonian Quantum Walks: From The Franck-Mittner-Devlin Work-In-Progress To Quantum Mechanics
- A Few-Qubit Generation Process From Zero Spin/3-Qubit States With Many Degrees Of Freedom
- Asymptotics And Entanglement Of Optomechanics In A Nonlinear Magnetic Field System
- Quantum Computing Of Quantum Entangled Electrons
- Theorems Of Quantum Mechanics
- A Supersquare Operator Based On The Operator And Nonlocality Of Momentum In A Few-Photon Qubit System
- A Single-Photon Hamiltonian From Schrödinger'S Group
- A Few-Qubit Generating Process From Unruh Phase Space
- Characterizing Symmetric Phases In Phase Space
- A Generalized Description Of Quantum Walks
- Analysis Of Bipartite Processes From Superconductors In A Gas-One Cavity
- Generalised Error Correction For Arbitrary Precision Errors$ 
```

Rys.3 Wyświetlenie tytułów wygenerowanych podczas uczenia.

### 3.3 Generowanie tytułu na podstawie wytrenowanego modelu

Po wytrenowaniu modelu istnieje możliwość wgrania go do GPT-2 w celu generowania nowych tytułów publikacji naukowych. Pierwszym krokiem jest wgranie zapisanego modelu z pomocą funkcji `load_gpt2()`. Gdy model jest już dostępny, używając funkcji `generate()` można wygenerować nowe tytuły.

Wygenerowane tytuły wyświetlono po wcześniejszym usunięciu niepotrzebnych fragmentów tekstu.

```

import gpt_2_simple as gpt2
import tensorflow as tf

tf.compat.v1.reset_default_graph()

sess = gpt2.start_tf_sess()

gpt2.load_gpt2(sess, reuse=tf.compat.v1.AUTO_REUSE)

```

Loading checkpoint checkpoint\run1\model-60  
INFO:tensorflow:Restoring parameters from checkpoint\run1\model-60

Rys.4 Załadowanie wytrenowanego modelu

```

prefix = 'neural' # None is default

text = gpt2.generate(sess,
    length=40,
    temperature=0.7,
    prefix=prefix,
    nsamples=1,
    batch_size=1,
    return_as_list=True
)

t = text[0].title()

for s in ['Endoftext', 'Startoftext', '<|', '|>', ']', '[', '|', '\n']:
    t = t.replace(s, '')
print(t)

```

Used To Be A Strong-State For Many Different Quantum StatesQuantum State Engineering For The Quantum Internet

Rys.5 Generowanie tytułów

## 4. Podsumowanie

Wyniki widoczne na rysunku 3 zawierają sensowne tytuły, które mogłyby być wykorzystane w celach naukowych. Niestety tytuły generowane za pomocą wytrenowanego modelu nierzadko nie mają sensu lub występują w nich powtórzenia. Mimo wszystko, w większości przypadków da się z powodzeniem, lub po drobnej zmianie, użyć wygenerowanego tytułu. Z pewnością na podstawie tych oraz kolejnych rezultatów można przewidywać kierunek rozwoju wybranej dziedziny oraz samego przetwarzania języka naturalnego.

## **5. Bibliografia**

1. Dokumentacja Arxivscraper - <https://github.com/Mahdisadjadi/arxivscraper>
2. Biblioteka ArXiv - [https://arxiv.org/category\\_taxonomy](https://arxiv.org/category_taxonomy)
3. Baza projektu - <https://github.com/csinva/gpt-paper-title-generator>