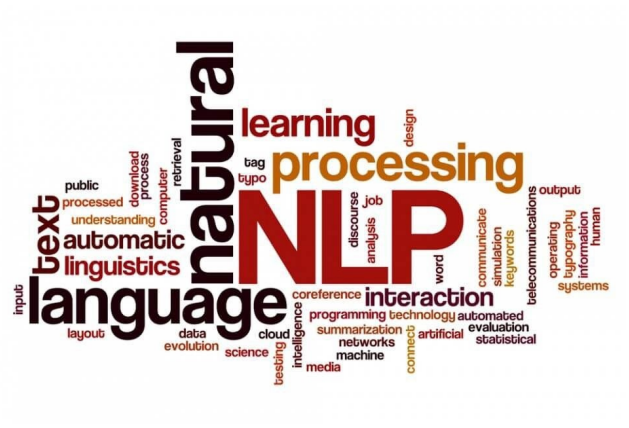
بسم الله الرحمن الرحیم




# دانشکده مهندسی برق و کامپیوتر

**پردازش زبان‌های طبیعی - تمرین ششم**

**سید مهدی رضوی**

استاد : آقای دکتر فیلی

تیر ماه ۱۴۰۳

# فهرست مطالب



# فهرست تصاویر



# ۱ بخش اول

RecursiveCharacterTextSplitter یک ابزار کاربردی در LangChain است که برای تقسیم اسناد طولانی به قطعات کوچکتر و قابل مدیریت طراحی شده است. این به ویژه در هنگام برخورد با متون بزرگی که نیاز به پردازش، نمایه سازی یا جستجو دارند، مفید است. در اینجا توضیح مفصلی در مورد chunksize و chunkoverlap آمده است:

: chunksize

تعریف: chunksize حداکثر تعداد کاراکترهایی را که هر تکه شامل می شود را مشخص می کند.

هدف: این پارامتر تضمین می کند که هر تکه از متن خیلی بزرگ نیست، که به دلایل مختلفی می تواند مفید باشد: عملکرد: تکه‌های کوچک‌تر پردازش آسان‌تر بوده و به حافظه کمتری نیاز دارند. ارتباط: به حفظ تمرکز و زمینه متن در محدوده‌های قابل مدیریت کمک می‌کند و کار مدل‌های زبان و الگوریتم‌های جستجو را آسان‌تر می‌کند. نمایه سازی: تکه های کوچکتر برای نمایه سازی و بازیابی در پایگاه های داده برداری مانند FAISS کارآمدتر هستند.

: chunkoverlap

تعریف: chunkoverlap تعداد کاراکترهایی را که باید بین تکه های متوالی همپوشانی داشته باشند را مشخص می کند. هدف: این پارامتر تضمین می‌کند که بین تکه‌ها همپوشانی وجود دارد که به دلایل مختلفی می‌تواند مفید باشد:

Preservation: Context به حفظ زمینه بین تکه ها کمک می کند. این مهم است زیرا برخی از اطلاعات مرتبط ممکن است در مرز یک تکه قطع شوند. تداوم: تضمین می‌کند که یک انتقال آرام بین تکه‌ها وجود دارد، و خطر از دست دادن زمینه معنادار را که ممکن است بین تکه‌ها تقسیم شود، کاهش می‌دهد.

Loading Data

```
import requests
from bs4 import BeautifulSoup

url = "https://stanford.edu/~jurafsky/slp3/"
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')

pdf_links = []
for link in soup.find_all('a', href=True):
    href = link['href']
    if href.endswith('.pdf'):
        pdf_links.append(url + href if href.startswith('/') else href)
```

```
[ ] pdf_links_main = []
    for pdf in pdf_links:
        item = 'https://' + append_text + pdf
        pdf_links_main.append(item)
        print(item)
```

Executing (2m 33s) <cell line: 5> › from_documents() › from_texts() › embed_documents() › encode

شکل ۱: بارگذاری تمام لینک‌ها در صفحه رسمی کتاب

## ۲ بخش دوم

در LangChain ، Embedder یک نقش کلیدی ، در در تبدیل متن به بردارهای عددی ایفا می‌کند. این بردار عددی که معنای متن را به تصویر می کشد.
این بردارهای عددی برای کارهای مختلف مانند جستجوی شباهت، خوشه‌بندی و بازیابی معنایی ضروری هستند.
Embedder سنگ بنای درک معنایی در LangChain است که سیستم را قادر می سازد از تطابق واژگانی فراتر رفته و از روابط معنایی عمیق بین متون استفاده کند. این منجر به نتایج دقیق‌تر، معنادارتر و مرتبط‌تر در کاربردهای مختلف می‌شود.

```
urls = [
    'https://stanford.edu/~jurafsky/slp3/2.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/3.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/4.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/5.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/6.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/7.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/8.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/9.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/10.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/11.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/12.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/13.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/14.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/15.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/16.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/17.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/18.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/19.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/20.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/21.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/22.pdf' ,
    'https://stanford.edu/~jurafsky/slp3/23.pdf' ,
]
```

شکل ۲: لینک فصل‌های مختلف کتاب ژورافسکی

Chunking, Vector Store and Retriever

```
text_splitter = RecursiveCharacterTextSplitter(chunk_size = 1024 , chunk_overlap = 64)
chunks = text_splitter.split_documents(docs)

embedding_function = HuggingFaceEmbeddings(show_progress = True, multi_process = True)
vector_store = FAISS.from_documents(documents=chunks, embedding = embedding_function)

retriever = vector_store.as_retriever(search_kwargs = {"k": 3})
```

```
/usr/local/lib/python3.10/dist-packages/langchain_core/_api/deprecation.py:139: LangChainDeprecationWarning: The class `HuggingFaceEmb
  warn_deprecated(
/usr/local/lib/python3.10/dist-packages/sentence_transformers/cross_encoder/CrossEncoder.py:11: TqdmExperimentalWarning: Using `tqdm.a
  from tqdm.autonotebook import tqdm, trange
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as sec
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
```

شکل ۳: عملیات قطعه‌بندی کردن متون

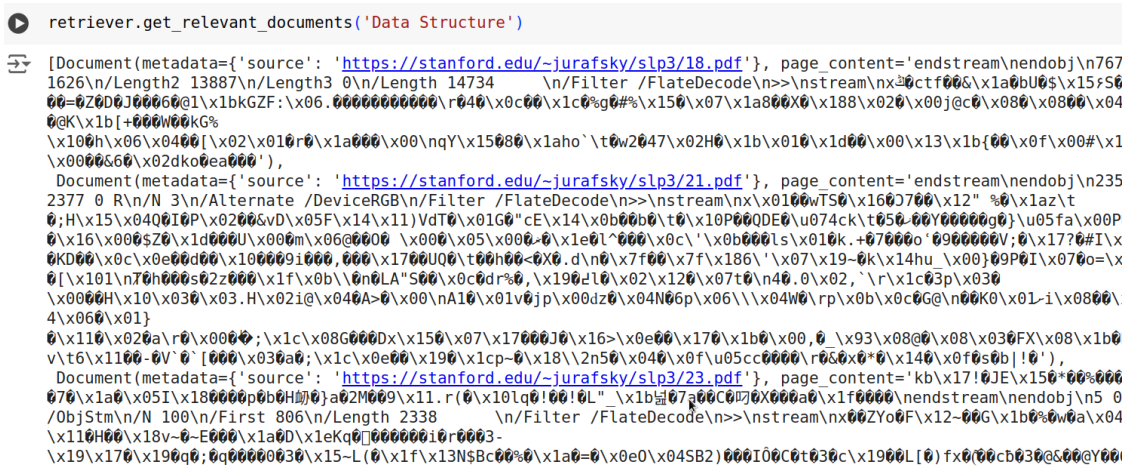
# ۳ بخش سوم

- رویکرد بازیاب معنایی بر مبنای Embedding-based matching
- می تواند مفهوم و مفهوم کلمات و جملات را درک کند.
- مترادف ها، نقل قول ها، و تغییرات در جمله بندی را به طور موثر مدیریت می کند.
- برای پرس و جوهای پیچیده که نیاز به درک معنایی دارند، موثرتر است.
- عموماً به دلیل هزینه محاسباتی تولید Embedding ، کندتر از بازیابی‌های واژگانی است.
- به مدل های از پیش آموزش دیده و گاهی اوقات منابع محاسباتی بزرگ نیاز دارد.
- برای دستیابی به عملکرد مطلوب ممکن است برای موارد استفاده خاص نیاز به تنظیم دقیق داشته باشد.

- رویکرد بازیاب واژگانی
- رویکرد: تطبیق کلمات کلیدی (Algorithms like TF-IDF)
- سریع و کارآمد برای پرس و جوهای ساده.
- زمانی که تطبیق دقیق کلمه کلیدی بسیار مهم است، به خوبی کار می کند.
- نیازی به آموزش یا تعبیه های از پیش محاسبه شده ندارد.
- نقاط ضعف: قادر به درک مفهوم یا مفهوم کلمات نیست.
- نمی تواند مترادف ها یا نقل قول ها را به طور موثر مدیریت کند.
- برای پرس و جوهای پیچیده که در آن معناشناسی اهمیت دارد، کمتر موثر است.


شکل ۴: اسناد بازگردانده‌شده توسط بازیاب در جواب یک پرس‌وجو مرتبط با پردازش زبان


شکل ۵: اسناد بازگردانده‌شده توسط بازیاب در جواب یک پرس‌وجو مرتبط با پردازش زبان




شکل ۶: اسناد بازگردانده‌شده توسط بازیاب در جواب یک پرس‌وجو مرتبط با علوم کامپیوتر

# ۴ بخش چهارم

```
retriever.get_relevant_documents('Data Structure')
```

```
[Document(metadata={'source': 'https://stanford.edu/~jurafsky/slp3/18.pdf'}, page_content='endstream\nendobj\n767
1626\n/Length2 13887\n/Length3 0\n/Length 14734     \n/Filter /FlateDecode\n>>\nstream\nx…@ctf@@&\x1a@bU@$\x15ŕS@
@@=@Z@D@J@@@6@@1\x1bkGZF:\x06.@@@@@@@@@@\r@4@\x0c@@\x1c@%g@#%\x15@\x07\x1a8@@X@\x188\x02@\x00j@c@\x08@\x08@@\x04
@@K\x1b[+@@@W@@kG%
\x10@h\x06\x04@@[\x02\x01@r@\x1a@@@\x00\nqY\x15@8@\x1aho`\t@w2@47\x02H@\x1b\x01@\x1d@@\x00\x13\x1b{@@\x0f\x00#\x1
\x00@@&6@\x02dko@ea@@@'),
 Document(metadata={'source': 'https://stanford.edu/~jurafsky/slp3/21.pdf'}, page_content='endstream\nendobj\n235
2377 0 R\n/N 3\n/Alternate /DeviceRGB\n/Filter /FlateDecode\n>>\nstream\nx\x01@@wTS@\x16@J7@@\x12" %@\x1az\t
@;H\x15\x04Q@I@P\x02@@&vD\x05F\x14\x11)VdT@\x01G@"cE\x14\x0b@@b@\t@\x10P@@QDE@\u074ck\t@5@~@@Y@@@@g@}\u05fa\x00P
@\x16\x00@$Z@\x1d@@@U\x00@m\x06@@@O@ \x00@\x05\x00@~@\x1e@l^@@@\x0c\'\x0b@@@ls\x01@k.+@7@@@o'@9@@@@@V;@\x17?@#I\x
@KD@@\x0c\x0e@@d@@\x10@@@9i@@@,@@@\x17@@UQ@\t@@h@@<@X@.d\n@\x7f@@\x7f\x186\'\x07\x19~@k\x14hu_\x00}@9P@I\x07@o=\x
@[\x101\n7@h@@@s@2z@@@\x1f\x0b\\@n@LA"S@@\x0c@dr%@,\x19@ل@\x02\x12@\x07t@\n4@.0\x02,`\r\x1c@3p\x03@
\x00@@H\x10\x03@\x03.H\x02i@\x04@A>@\x00\nA1@\x01v@jp\x00dz@\x04N@6p\x06\\\x04W@\rp\x0b\x0c@G@\n@@K0\x01~i\x08@@\
4\x06@\x01}
@\x11@\x02@a\r@\x00@@;\x1c\x08G@@@Dx\x15@\x07\x17@@@J@\x16>\x0e@@\x17@\x1b@\x00,@_\x93\x08@@\x08\x03@FX\x08\x1b@
v\t6\x11@@-@V`@`[@@@\x03@a@;\x1c\x0e@@\x19@\x1cp~@\x18\\2n5@\x04@\x0f\u05cc@@@@\r@&@x@*@\x14@\x0f@s@b|!@'),
 Document(metadata={'source': 'https://stanford.edu/~jurafsky/slp3/23.pdf'}, page_content='kb\x17!@JE\x15@*@@%@@@
@7@\x1a@\x05I\x18@@@p@b@H@@}a@2M@@9\x11.r(@\x10lq@!@@!@L"_\x1b뉲@7a@@C@@@X@@@a@\x1f@@@@\nendobj\nendobj\n5 0
/ObjStm\n/N 100\n/First 806\n/Length 2338     \n/Filter /FlateDecode\n>>\nstream\nx@@ZYo@F\x12~@@G\x1b@%@w@a\x04
\x11@H@@\x18v~@~E@@@\x1a@D\x1eKq@□@@@@@@i@r@@@3-
\x19\x17@\x19@q@;@q@@@@@3@\x15~L(@\x1f\x13N$Bc@@%@\x1a@=@\x0eO\x04SB2)@@@IÔ@C@t@3@c\x19@@L[@)fx@(@@cb@3@@&@@@Y@@
```

شکل ۷: سناد بازگردانده‌شده توسط بازیاب در جواب یک پرس‌وجو غیرمرتبط

```
[ ] raw_prompt3 = 'What is NLP Reference book and What\'s the topics of this book ? please tell me in persian . thanks'
    llm.invoke(raw_prompt3).content
```

```
'Here is the information you requested:\n\n**NLP Reference Book:**\n\nOne of the most popular and widely used NLP reference books is "Nat
ural Language Processing (almost) from Scratch" by Collobert et al. (2011). However, there are many other excellent books on NLP that can
serve as a reference, depending on your specific needs and goals.\n\n**Topics Covered in a Typical NLP Reference Book:**\n\nHere are some
of the typical topics covered in a comprehensive NLP reference book:\n\n1. **Mathematical Foundations**: Linear Algebra, Calculus, Probab
ility Theory, and Information Theory.\n2. **Language Modeling**: N-gram models, Markov models, and neural network-based language model
s.\n3. **Text Preprocessing**: Tokenization, Stopword removal, Stemming, Lemmatization, and Named Entity Recognition.\n4. **Word Embeddin
gs**: Word2Vec, GloVe, and FastText.\n5. **Text Classification**: Sentiment Analysis, Spam Detection, and Topic Modeling.\n6. **Sequence
Labeling**: Part-of-Speech Tagging, Named E…'
```

```
prompt_template = "Hi, I'm learning {tool} to use it for classify of {query} in one of the 3 categories. \
Natural Language Processing , Computer Science or None of them."

prompt_chat_template = ChatPromptTemplate.from_template(
    template=prompt_template,
)

prompt_chat_template.invoke({
    "tool":"llama_index" ,
    "query" : "query"
})
```

```
----------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
```

0s completed at 11:35 AM

شکل ۸: ChatPromptTemplate

## Section 4

```
llm = ChatTogether(model_name="meta-llama/Llama-3-70b-chat-hf", temperature=0)

prompt_template = ChatPromptTemplate(
    template="Determine the context of the following query: {query}\n"
             "Is it related to NLP, Computer Science, or neither?",
    input_variables=["query"]
)

def get_context_chain(query):
    prompt = prompt_template.format(query=query)
    response = llm(prompt)
    return response

query = "Explain the transformer architecture used in NLP."
context = get_context_chain(query)
print(f"Query: {query}")
print(f"Context: {context}")
```

شکل ۹: ChatPromptTemplate

```
from pprint import pprint

# raw_prompt = "Hi, I'm learning langchain to use it with LLMs and make awesome stuff!"
# raw_prompt = "سلام . من علاقه‌مند به یادگیری تکنولوژی اسپارک و هدوپ و کافکا برای یافتن شغل مهندس داده هستم."

raw_prompt2 = 'Hi my dear assistant ! What is NLP and What does it do in our lifes ?'
raw_prompt3 = 'What is NLP Reference book and What\'s the topics of this book ? please tell me in persian . thanks'
llm.invoke(raw_prompt3).content
```

```
'Here is the information you requested:\n\n**NLP Reference Book:**\n\nOne of the most popular and widely used NLP reference books is "Nat
ural Language Processing (almost) from Scratch" by Collobert et al. (2011). However, there are many other excellent books on NLP that can
serve as a reference, depending on your specific needs and goals.\n\n**Topics Covered in a Typical NLP Reference Book:**\n\nHere are some
of the typical topics covered in a comprehensive NLP reference book:\n\n1. **Mathematical Foundations**: Linear Algebra, Calculus, Probab
ility Theory, and Information Theory.\n2. **Language Modeling**: N-gram models, Markov models, and neural network-based language model
s.\n3. **Text Preprocessing**: Tokenization, Stopword removal, Stemming, Lemmatization, and Named Entity Recognition.\n4. **Word Embeddin
gs**: Word2Vec, GloVe, and FastText.\n5. **Text Classification**: Sentiment Analysis, Spam Detection, and Topic Modeling.\n6. **Sequence
Labeling**: Part-of-Speech Tagging, Named E...'
```

```
[ ] prompt_template = "Hi, I'm learning {tool} to use it for classify of {query} in one of the 3 categories. \
    Natural Language Processing , Computer Science or None of them."

    prompt_chat_template = ChatPromptTemplate.from_template(
        template=prompt_template,
    )
```

شکل ۱۰: llm invoke