به نام خدا

علی عزیزی - 9731042

پروژه سوم درس هوش مصنوعی

معرفی پروژه

در این پروژه، مدل های زبانی unigram و bigram برای سه پیکره زبانی مختلف که متعلق به فردوسی، حافظ و مولانا، سه شاعر بزرگ ایرانی هستند ساخته شده تا ساختار زبانی هر یک از این شعرا را مدل کند.
در مرحله بعد، برنامه پس از دریافت یک مصرع جدید از پیکره ی تست، می بایست تشخیص دهد که کدام یک از این سه شاعر بزرگ آن مصرع را سروده است.

شرح پیاده سازی انجام شده

در این قسمت به شرح توابع مهم برنامه و روند رسیدن برنامه از ورودی به خروجی می پردازیم.

read_input(path):

در این تابع مصرع های یک شاعر از یک فایل تکست خوانده شده و لیستی از مصرع های ویرایش شده و یک واژه نامه از لغاتی که شاعر در مصرع های خود از آن استفاده کرده است،به عنوان خروجی برگردانده می شود.

edit_sentences(sentences: List[str]):

این تابع لیستی از مصرع ها را به عنوان ورودی می گیرد و در یک حلقه علائم نگارشی هر مصرع را حذف می کند و به ابتدا و انتهای هر مصرع کلمات مشخصی را اضافه می کند تا کلمات ابتدایی و انتهایی مشخص شوند.

unkonwn_finder(sentences: List[str]):

این تابع کلماتی که کمتر از دو بار در متن تکرار شده اند را با کلمه "<ناشناخته>" جایگزین می کند.

generate_unigram(sentences: List[str])
generate_bigram(sentences: List[str])

این دو تابع لیستی از مصرع ها را به عنوان ورودی می گیرند و لیستی از unigram ها و bigram ها را به عنوان خروجی بر میگردانند.

generate_bigram_for_test(sentence: str)

این تابع یک مصرع به عنوان ورودی می گیرد و توالی های دو تایی را در لیستی به عنوان خروجی بر می گرداند، از این تابع در تابع محاسبه backoff model یک مصرع استفاده می شود.

learn(sentences: List[str])

در این تابع مدل های unigram و bigram برای یک مصرع های تولید می شود و در واقع مدل های احتمالاتی که هنگام تست از آن ها استفاده می شود را تولید می کند.

backoff_model(unigram, bigram, words)

در این تابع مدل backoff برای یک توالی دو تایی مطابق فرمول آورده شده در دستور کار تولید میشود.

read_test_set()

این تابع وظیفه دارد پیکره تست را از فایل تکست بخواند.

روند تولید خروجی

ابتدا مصرع هایی که هر شاعر آن ها را سروده است، از فایل ورودی خوانده شده و برای هر کدام از شاعر ها مدل های زبانی ذکر شده ساخته می شود، سپس در یک قطعه کد برای هر شاعر تعداد مصرع هایی که درست تشخیص داده شده اند توسط یک کانتر شمارش می شود که به عنوان مثال این قطعه کد برای حافظ در ادامه آورده شده است.

```
hafez_correct = 0
for sentence in hafez_test_sentences:
    res_f = 1
    res_h = 1
    res_m = 1
    for k in generate_bigram_for_test(sentence):
        res_f *= backoff_model(ferdowsi_unigram, ferdowsi_bigram, k)
        res_h *= backoff_model(hafez_unigram, hafez_bigram, k)
        res_m *= backoff_model(molana_unigram, molana_bigram, k)
    if max(res_m, res_h, res_f) == res_h:
        hafez_correct += 1
```

خروجی

به ازای λ و ε های مختلف نتایج متفاوتی حاصل می شود که در ادامه چهار مورد از آن های آورده شده است.

حالت اول:

$\lambda_3 = 0.9$

$\lambda_2 = 0.09$

$\lambda_1 = 0.01$

$\varepsilon = 0.0001$

```
Complete Ferdowsi training ...
Complete Hafez training ...
Complete Molana training ...
Ferdowsi :  91.60000000000001 %
Hafez :  85.38011695906432 %
Molana :  76.49812734082397 %
Overall : 84.19331395348837 %
```

حالت دوم:

$\lambda_3 = 0.1$

$\lambda_2 = 0.2$

$\lambda_1 = 0.7$

$\varepsilon = 0.0001$

```
Complete Ferdowsi training ...
Complete Hafez training ...
Complete Molana training ...
Ferdowsi :  90.5 %
Hafez :  82.01754385964912 %
Molana :  69.5692883895131 %
Overall : 80.2688953488372 %
```

حالت سوم:

$\lambda_3 = 0.15$

$\lambda_2 = 0.7$

$\lambda_1 = 0.15$

$\varepsilon = 0.0001$

```
Complete Ferdowsi training ...
Complete Hafez training ...
Complete Molana training ...
Ferdowsi :  91.7 %
Hafez :  82.16374269005848 %
Molana :  71.34831460674157 %
Overall : 81.43168604651163 %
```

حالت چهارم :

$\lambda_3 = 0.6$

$\lambda_2 = 0.2$

$\lambda_1 = 0.2$

$\varepsilon = 0.05$

```
Ferdowsi :  84.39999999999999 %
Hafez :  68.42105263157895 %
Molana :  68.91385767790263 %
Overall : 74.4186046511628 %
```

حالت پنجم:

$\lambda_3 = 0.1$

$\lambda_2 = 0.15$

$\lambda_1 = 0.75$

$\varepsilon = 0.05$

```
Ferdowsi :  81.6 %
Hafez :  62.71929824561403 %
Molana :  57.67790262172284 %
Overall : 67.62354651162791 %
```

حالت ششم:

$\lambda_3 = 0.15$

$\lambda_2 = 0.7$

$\lambda_1 = 0.15$

$\varepsilon = 0.05$

```
Ferdowsi :  81.8 %
Hafez :  70.76023391812866 %
Molana :  57.39700374531835 %
Overall : 69.58575581395348 %
```

نتیجه گیری نهایی

دقت تشخیص برنامه به پارامتر های لاندا و اپسیلون وابسته است و همانطور که در شش نمونه خروجی فوق نشان داده است با تغییر این پارامتر ها دقت تشخیص برنامه دستخوش تغییرات می شود.

برای یافتن بهترین مقادیر موجود برای این پارامتر ها می بایست از الگوریتم های یادگیری ماشین استفاده نمود تا دقت برنامه بیشینه شود اما توجه به خروجی های تولید شده می توان با تقریب خوبی گفت که بالاترین دقت خروجی زمان تولید می شود که ضریب bigram در فرمول backoff model از دیگر ضرایب لاندا کمتر باشد و مقدار اپسیلون نیز به اندازه کافی کم باشد دلیل **بالا بودن ضریب bigram** این است که می بایست برای توالی های دوتایی که در پیکره تست هستند ضریب بیشتری نسبت به تک کلمه ها و همچنین ضریب اپسیلون قائل شد. دلیل اینکه باید **اپسیلون به اندازه کافی کوچک باشد** این است که ممکن است عدد که برای حاصلضرب $\varepsilon \times \lambda_1$ بدست می آید به حدی بزرگ شود که نقش توالی دو تایی (bigram) نادیده گرفته شود.