دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گزارش پروژه سوم

مبانی و کاربردهای هوش مصنوعی

نگارش

آرش حاجی صفی - ۹۶۳۱۰۱۹

بهمن ۱۳۹۹

توضیح کد نوشته شده برای حل مسئله:

الگوریتم‌های مورد نیاز برای ساخت مدل‌های زبانی و محاسبه احتمالات unigram و bigram را در فایل nlp_algorithms.py پیاده‌سازی نموده‌ام و در فایل poet_identification.py با استفاده از این توابع، مسئله را حل نموده‌ام.

یک کلاس به اسم Language را نوشته‌ام که شامل فیلدهای file_name (اسم فایل training set مورد نظر برای زبان مورد نظر)، words_dict (یک دیکشنری از تمامی واژه‌هایی که در traning set زبان مورد نظر قرار داشته‌اند به همراه تعداد دفعات تکرارشان)، unigram_model (یک دیکشنری که هر کلمه از مجموعه آموزشی را به احتمال unigram اش بدون هیچ backoff modelای نگاشت می‌کند)، و bigram_model (یک دیکشنری که هر کلمه دوتایی به صورت "word2-word1" را به احتمال bigram وقوع word2 به شرط اینکه word1 قبل از آن آمده باشد بدون هیچ backoff modelای نگاشت می‌کند) می‌باشد.

برای هر یک از این سه شاعر، یک شیء از این کلاس می‌سازیم. در این کلاس تابع init_dict را پیاده‌سازی نموده ام که واژه نامه از تمامی لغات موجود در مجموعه آموزشی که برای آن شیء را ساخته‌ایم می‌سازد (در ضمن واژه‌هایی را که تعداد دفعات آنها کمتر از پارامتر سراسری WORDS_THRESHOLD است، از واژه نامه حذف می‌کند). تابع build_bigram_model و build_unigram_model نیز بر روی هر یک از اشیاء کلاس Language صدا زده می‌شود و مدل‌های bigram و unigram یادشده را برای آنها مطابق فرمتی که گفته شد، می‌سازد. پس برای این سه شاعر تمامی این موارد ساخته می‌شود.

همچنین در کلاس Language، تابع bigram_prob را نوشته ام که ۲ کلمه word2 و word1 را می‌گیرد و احتمال bigram برای وقوع word1 به شرط اینکه قبل از آن در متن word2 آمده باشد را با استفاده از backoff model گفته شده در صورت سوال برای زبان شاعر گفته شده محاسبه می‌کند و برمی‌گرداند.

در فایل poet_identification.py پس از اینکه این سه مجموعه زبانی برای شاعرها ساخته شد و مدل‌های هر کدام ایجاد شد، آنها را به صورت یک آرایه به تابع predict_labels پاس می‌دهم. تابع predict_labels آدرس فایل تست و آرایه ای از این شاعرها را که اشیائی از جنس Language هستند را می‌گیرد و به ازای هر مصراع شعر از مجموعه تست، تابع predict را بر روی این آرایه از شاعرها و جمله مورد نظر صدا می‌زند. تابع predict احتمال وقوع جمله مورد نظر را با استفاده از ضرب احتمالات backoff model برای هر شاعر حساب می‌کند و نهایتاً این احتمال برای هرکدام که بیشتر شد، label آن را به عنوان label پیش‌بینی شده بر می‌گرداند و این مصراع شعر را متعلق به شاعر با این label تشخیص می‌دهد. نهایتاً تمامی لیبل‌های پیش‌بینی شده که برگردانده می‌شوند با تمامی لیبل‌های واقعی موجود در فایل تست نظیر به نظیر مقایسه می‌شوند و تعداد حدس‌های درست و نسبت حدس‌های درست به تمامی حدس‌ها را با تابع calculate_accuracy بدست می‌آوریم که همان دقت تشخیص برنامه ما می‌باشد.

گزارشی از نتایجی که در صورت سوال خواسته شده نیز در صفحه بعد آورده شده است.

گزارش نتایج:

حد آستانه تکرار کلمات برای حذف از دیکشنری مورد نظر را ۲ در نظر گرفته ایم (کلماتی که ۲ یا تعداد دفعات کمتری تکرار شده‌باشند حذف از واژه‌نامه حذف شده‌اند).

تست به ازای دو حالت برای ε:

تست اول:

پارامترها: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.0001

دقت مدل: 84.4%

```
TERMINAL  DEBUG CONSOLE  PROBLEMS 11  OUTPUT

PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/arash/AppData/Local/P
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.0001

Accuracy of the predicted model: 0.8441133720930233

All predictions: 2752
Correct predictions: 2323
```

تست دوم:

پارامترها: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.1

دقت مدل: 78.3%

```
TERMINAL  DEBUG CONSOLE  PROBLEMS 11  OUTPUT

PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/arash
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.1

Accuracy of the predicted model: 0.7830668604651163

All predictions: 2752
Correct predictions: 2155
```

مقادیر λ همان مقادیر تست اول هستند و تنها مقدار ε را از 0.0001 به 0.1 افزایش دادیم که می‌بینیم دقت بدتر شد! علت این است که با افزایش ε مقدار عبارت λ1*ε در backoff model زیاد می‌شود و از آنجایی که این عبارت هیچ اطلاعاتی را در خصوص احتمال وقوع یک کلمه یا دو کلمه پشت سر هم (unigram و bigram) را به ما نمی‌دهد و مقدار آن برای همه‌ی عبارت‌ها ثابت

است، هیچ کمکی به ما نمی‌کند و هرچه بزرگتر شود نقش احتمال bigram و unigram که اطلاعات بسیار مهم‌تری را به ما می‌دهند کمتر می‌شود و در نتیجه دقت پایین می‌آید.

تست به ازای دو حالت برای λ :

تست سوم:

پارامترها: λ3: 0.3, λ2: 0.4, λ1: 0.3, ε: 0.0001

دقت مدل: 83.7%

```
PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/ara
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.3, λ2: 0.4, λ1: 0.3, ε: 0.0001

Accuracy of the predicted model: 0.8379360465116279

All predictions: 2752
Correct predictions: 2306
```

تست چهارم:

پارامترها: λ3: 0.75, λ2: 0.27, λ1: 0.03, ε: 0.0001

دقت مدل: 84.3%

```
TERMINAL  DEBUG CONSOLE  PROBLEMS 11  OUTPUT

PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/arash
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.75, λ2: 0.27, λ1: 0.03, ε: 0.0001

Accuracy of the predicted model: 0.8430232558139535

All predictions: 2752
Correct predictions: 2320
```

می‌بینیم که با ثابت نگه‌داشتن مقدار ε، هرچه مقدار λ3 را بیشتر کنیم و مقدار λ1 را از همه کمتر کنیم و مقداری بین این دو را به λ2 دهیم، دقت مدل بالاتر می‌رود.

علت این است که λ3 در احتمال bigram ضرب می‌شود که هرچه این احتمال بیشتر باشد، احتمال اینکه عبارت دو کلمه ای مورد نظر مربوط به زبان مورد نظر باشد بالاتر است و bigram دقیق‌تر از unigram عمل می‌کند. در ضمن λ1 نیز دوباره در ε ضرب می‌شود که هیچ اطلاعات خاصی را به ما نمی‌دهد و صرفاً برای صفر نشدن احتمال مورد نظر جمع می‌شود و در نتیجه هرچه کمتر باشد نتیجه بهتری می‌گیریم. λ2 نیز در احتمال unigram کلمه ضرب می‌شود و خوب است که مقداری بسیار بیشتر از λ1 و تا حدی کمتر از λ2 را داشته باشد تا به نتیجه خوبی برسیم؛ چون احتمال وقوع صرفاً یک کلمه نه با مانند unigram احتمال اینکه جمله متعلق به زبان مورد نظر باشد را باید خیلی بالا ببرد و نه مانند ضرب λ1 در ε است که هیچ کارایی نداشته باشد و اطلاعات مفیدی را به ما می‌دهد و منطقاً هرچه زیادتر باشد تا حدی باید احتمال تعلق جمله به زبان مورد نظر را بالاتر ببرد و در نتیجه خوب است که مقداری مابین این دو برایش در نظر گرفته شود.

بهترین پارامترها و علت برتری:

با توجه به تست‌هایی که صورت گرفت، بهترین پارامترها به شرح زیر می‌باشند:

`λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.0001`

دقت مورد نظر در این حالت برابر با 86% (در حالتی که هیچ کلمه ای را از واژه نامه حذف نکنیم)، و 84.4% (در حالتی که حد تعداد دفعات آستانه برای حذف کلمه از واژه نامه را ۲ قرار دهیم) می‌باشد.

حالت بدون حذف لغت از واژه نامه:

```
TERMINAL   DEBUG CONSOLE   PROBLEMS 11   OUTPUT

PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.0001

Accuracy of the predicted model: 0.8604651162790697

All predictions: 2752
Correct predictions: 2368
```

حالت قرار دادن حذف آستانه برابر ۲ برای حذف لغت از واژه‌نامه:

```
TERMINAL  DEBUG CONSOLE  PROBLEMS 11  OUTPUT

PS K:\Bachelor\Principles of Artificial Intelligence\Project 3\final\code> & C:/Users/arash/AppDa
les of Artificial Intelligence/Project 3/final/code/poet_identification.py"

Finished!

Parameters: λ3: 0.7, λ2: 0.25, λ1: 0.05, ε: 0.0001

Accuracy of the predicted model: 0.8441133720930233

All predictions: 2752
Correct predictions: 2323
```

علت برتری نیز همانطور که در قسمت‌های قبلی گفته شد، این است که در بهترین حالت، باید برای $\lambda 3$ مقداری بزرگتر از همه در نظر گرفته شود، چرا که در احتمال bigram ضرب می‌شود و این احتمال، کارکرد خیلی خوبی را نسبت به بقیه عبارت‌ها برای تشخیص اینکه آیا عبارت متعلق به زبان مورد نظر است یا نه می‌دهد و باید بیشترین تاثیر را داشته باشد. سپس باید $\lambda 2$ مقداری کمتر از $\lambda 3$ و بیشتر از $\lambda 1$ را داشته باشد، چراکه احتمال unigram نه خیلی قابل اتکا است (نسبت به bigram) و نه کاملاً بلا استفاده است. مقدار ضرب $\lambda 1 \times \varepsilon$ نیز باید خیلی کوچک باشد چرا که هیچ کارکردی در خصوص اینکه آیا عبارت مربوط به یک زبان است یا نه را نمی‌دهد و صرفاً برای صفر نشن احتمال کل جمله آنرا در نظر می‌گیریم و باید انقدر کوچک باشد که تاثیری روی نتیجه احتمال bigram و نتیجه احتمال unigram در مجموع نداشته باشد. پس $\lambda 1$ و ε حاصل ضربشان باید خیلی کوچک شود.

در مجموع می‌توان گفت این λ ها طوری باید انتخاب شوند که احتمال bigram بیشترین تاثیر را در عبارت جمله مورد نظر داشته باشد و بعد از آن احتمال unigram تاثیر متعادلی را در احتمال بدست آمده برای کل جمله داشته باشد و حاصل ضرب $\lambda 1$ در ε کمترین تاثیر ممکن را داشته باشد.