دانشکده مهندسی کامپیوتر

تشخیص پروپاگاندا با استفاده از روش‌های یادگیری تقویتی

پایان‌نامه برای دریافت درجه کارشناسی در رشته مهندسی کامپیوتر

زهرا بشیر

استاد راهنما:

دکتر صالح اعتمادی

پاییز 99

تأییدیه ی هیأت داوران جلسه ی دفاع از پایان نامه

نام دانشکده: مهندسی کامپیوتر

نام دانشجو: زهرا بشیر

عنوان: تشخیص پروپاگاندا با استفاده از روش‌های یادگیری تقویتی

تاریخ دفاع: آذر 99

رشته: مهندسی کامپیوتر

ردیف	سمت	نام و نام خانوادگی	مرتبه دانشگاهی	دانشگاه یا موسسه	امضا
1	استاد راهنما	دکتر سید صالح اعتمادی	استادیار	دانشگاه علم و صنعت ایران	
2	استاد داور	دکتر محمدطاهر پیله ور	استادیار	دانشگاه علم و صنعت ایران	

تأییدیه‌ی صحت و اصالت نتایج

باسمه تعالی

اینجانب زهرا بشیر به شماره دانشجویی 95521072 دانشجوی رشته مهندسی کامپیوتر مقطع تحصیلی کارشناسی تایید می‌نمایم که کلیه‌ی نتایج این پایان‌نامه حاصل کار اینجانب و بدون هرگونه دخل و تصرف است و موارد نسخه‌برداری‌شده از آثار دیگران را با ذکر کامل مشخصات منبع ذکر کرده‌ام. در صورت اثبات خلاف مندرجات فوق، به تشخیص دانشگاه مطابق با ضوابط و مقررات حاکم (قانون جمایت از حقوق مولفان و مصنفان و قانون ترجمه و تکثیر کتب و نشریات و آثار صوتی، ضوابط و مقررات آموزشی، پژوهشی و انضباطی) با اینجانب رفتار خواهد شد و حق هرگونه اعتراض در خصوص احقاق حقوق مکتسب و تشخیص و تعییین تخلف و مجازات را از خویش سلب می‌نمایم. درضمن، مسئولیت هرگونه پاسخی به اشخاص اعم از حقیقی و حقوقی و مراجع ذی‌صلاح (اعم از اداری و قضایی) به عده‌ی اینجانب خواهد بود و دانشگاه هیچ‌گونه مسوولیتی در این خصوص نخواهد داشت.

نام و نام خانوادگی: زهرا بشیر

تاریخ و امضا:

مجوز بهره برداری از پایان نامه

بهره برداری از این پایان نامه در چهارچوب مقررات کتابخانه و با توجه به محدودیتی که توسط استاد راهنما به شرح زیر تعیین می‌شود، بلامانع است:

بهره برداری از این پایان نامه برای همگان بلامانع است.

بهره برداری از این پایان نامه با اخذ مجوز از استاد راهنما، بلامانع است.

بهره برداری از این پایان نامه تا تاریخ ..... ممنوع است.

نام استاد راهنما: دکتر سید صالح اعتمادی

تاریخ:

امضا:

چکیده

پروپاگاندا عبارتست از کوششی برای ترویج نظرات خاص سیاسی، اجتماعی، فرهنگی و ... از طریق بیان واقعیات با گزارش‌های غیر واقعی به قصد تاثیر گذاشتن بر ذهن مخاطب و ترغیب به رفتار خاص. در این میان، رسانه‌ها، فضای مجازی و سخنرانی‌ها همگی نقش مهمی در پراکندن این پروپاگانداها و تاثیر آن روی مردم دارند، بنابراین تشخیص پروپاگاندا از اهمیت ویژه ای برخوردار است. در این مقاله، به بررسی اینکه هر فرد در چه زمینه ای بیشتر پروپاگاندا تولید میکند میپردازد. (با توجه به سخنرانی ها و متونی که شخص از خودش در فضای مجازی منتشر می کند). این مساله در حالت عادی زمانگیر و غیر بهینه بوده و مزیتی که این مقاله نسبت به سایرین دارد، استفاده از تکنیک‌های یادگیری تقویتی در آن است. هدف بخش یادگیری تقویتی ما کم کردن تعداد موارد مورد بررسی و رسیدن به جواب با خواندن حداقل داده از شخص مورد بحث می باشد.[1]

فهرست مطالب

فهرست تصاویر

1 مقدمه

در طول تاریخ بشریت، ما شاهد چندین واقعه بوده ایم که باعث درد و سختی مردم در سراسر جهان شده است. حذف و جابجایی افراد بومی در آمریکای شمالی ، هولوکاست یهودیان جنگ جهانی دوم و پاک‌سازی قومی توتسیها در رواندا در اوایل دهه ۱۹۹۰ فقط چند نمونه است.

اما نکته جالب این است که در حالی که هر یک از این‌ها بدترین جنبه‌های بشریت را نشان می‌دهد، نمونه‌ای از استفاده موفق از پروپاگاندا هستند. پروپاگاندا یک شیوه ارتباطی است که برای دستکاری یا تأثیرگذاری بر عقاید گروه‌ها و یا برای پشتیبانی از یک علت یا عقیده خاص استفاده می‌شود. در طول قرن‌ها، پروپاگاندا به شکل هنری، فیلم، گفتار و موسیقی شکل گرفته‌است ، هرچند که محدود به این‌اشکال ارتباطی نیست. گرچه استفاده از پروپاگاندا منحصراً منفی نیست، اما اغلب مستلزم تأکید زیاد بر فواید و فضائل یک ایده یا گروه است، در عین حال تحریف همزمان حقیقت یا سرکوب ضد استدلال می‌باشد. به عنوان مثال، حزب نازی با ترویج این‌ایده که آلمان را دچار افسردگی اقتصادی کرد، قدرت گرفت.

همانطور که قبلاً بیان شد، پروپاگاندا به دلایل مختلف در زمان‌های مختلف استفاده می‌شود و به اشکال متنوعی قابل بررسی است. بنابراین اهمیت این‌موضوع و تشخیص آن بر همگان واضح است. همانطور که بالاتر ذکر شد هدف این‌مقاله بررسی زمینه غالب تولید پروپاگاندا در افراد با توجه به متن‌های صحبت‌هایشان است.

تصور کنید می‌خواهیم این‌موضوع که فردی در چه زمینه‌ای بیشتر پروپاگاندا تولید می‌کند را مورد بررسی قرار دهیم، در حالت عادی باید مجموعه‌ای از صحبت هایش (فرضا توییت) را مورد بررسی قرار دهیم، و پس از تعیین موضوع توییت، تشخیص دهیم آیا پروپاگاندا دارد یا نه. نتیجه را یادداشت کرده و برای هرزمینه نسبت تعداد پروپاگاندا به کل توییت‌ها را به دست آورده و به این‌ترتیب زمینه غالب آن‌فرد مشخص خواهد شد. همانطور که ملاحظه می کنید این‌کار به انرژی و وقت بسیار زیادی نیاز دارد. تلاش این‌مقاله اتوماتیک کردن این‌پروسه و همچنین کوتاه کردن رسیدن به پاسخ درست با استفاده از تکنیک‌های یادگیری تقویتی است.

بنابراین در این گزارش سه بخش اصلی پیاده سازی شده است:

- بخش تشخیص موضوع توییت
- بخش تشخیص وجود یا عدم وجود پروپاگاندا
- بخش یادگیری تقویتی

برای هر یک از این بخش‌ها پیاده سازی‌هایی با روش‌های متنوعی امتحان شده تا به مدل مطلوب‌تری برسیم. در بخش مدل پیشنهاد شده و آزمایش‌ها به جزئیات آن‌ها میپردازیم.

2 پیش‌زمینه

2.1 آشنایی با یادگیری تقویتی1

یادگیری تقویتی ، آموزش مدل‌های یادگیری ماشین برای تصمیم‌گیری متوالی است.

مساله ان آرم بندیت[2]:

این مسائل از ساده ترین مسائل یادگیری تقویتی هستند که در آنها یک عامل داریم که به آن‌عامل اجازه انتخاب عملیات[3] می دهیم، و هراکشن پاداشی[4] دارد که مطابق با یک تابع توزیع احتمال داده خواهد شد. این روند روی اپیزودهای زیادی اجرا می‌شود، و هدفش بیشینه کردن پاداش است.

هر یک از بازوها را که انتخاب کنید، پاداشی مطابق با آن دریافت می‌کنید. بنابراین به شما فرصتی داده می‌شود تا استراتژی خود را به نوعی توسعه دهید که بیشترین امتیاز را نهایتا دریافت کنید. موضوع بعدی که در مسائل یادگیری تقویتی اهمیت پیدا می‌کند، دوراهی کاوش- بهره برداری[5] است. اینکه تعداد داده‌های جدیدتری را ببینیم یا از داده‌های قبلی که تجربه کرده ایم استفاده کنیم.[2]

3 کارهای مرتبط

در زمینه تشخیص پروپاگاندا چندین کار کوچک شروع شده است و این موضوع از موضوعاتی است که هنوز با درصد بالایی حل نشده است. اما مقالات مربوط به تشخیص اخبار جعلی[6] تا حدی به بخش پردازش زبان طبیعی این پروژه نزدیک است. هم‌چنین کارگاه Sem Eval 2020 [3] که چندین ماه پیش برگزار شد، این موضوع را به عنوان یکی از چالش‌هایش مطرح کرده و تعدادی از تیم‌های شرکت‌کننده روی آن وقت گذاشتند و کدهایی را منتشر کردند.

به طور کلی در زمینه کاربرد یادگیری تقویتی در تشخیص جبهه اشخاص در تولید پروپاگاندا تاکنون مقاله یا کار مشابهی رویت نشده‌است.

---

[1] Reinforcement Learning
[2] N-arm bandit problem
[3] Action
[4] Reward
[5] Exploration-Exploitation dilemma
[6] Fake News

4 داده‌ها و مدل پیشنهاد شده

بخش تشخیص موضوع توییت: از یک مدل نایو بیز[7] برای تشخیص موضوع توییت (سیاسی – ورزشی – تفریحی – علمی – استارتاپ و...) استفاده شده است.

توجه داشته باشید که این مدل نایو بیز یک مدل پایه‌ای[8] و بسیار ساده است. در کارهای آینده قصد بهبود بخشی این مدل را دارم. دادگان این بخش از سایت track my hashtags[4] جمع‌آوری کرده‌ام.

بخش تشخیص پروپاگاندا: استفاده از یک شبکه عصبی عمیق[9] با به استفاده از امبدینگ کلمات[10] و چندین لایه LSTM. طبیعتا این یک روش پایه‌ای است و برای بهبود آن نهایتا از یک مدل برت[11] برای تشخیص پروپاگاندا دار بودن یا نبودن توییت‌ها استفاده خواهد شد.

دادگان آموزشی و آزمون این بخش را از همان سایت مربوط به کارگاه Sem Eval 2020 جمع‌آوری و مورد استفاده قرار دادم.

بخش یادگیری تقویتی: مساله ۷-آرم بندیت که با استفاده از روش نمونه برداری تامپسون پیاده سازی شده است.

برای دادگان این بخش به جهت تست بخش یادگیری تقویتی از همان سایتی که توییت‌های اشخاص را در اختیار می‌گذاشت، استفاده کردم.

5 پیاده سازی

این پروژه در لینک زیر قرار دارد:

https://gitlab.com/zahra_b77/nrlp-propaganda-recognition

پوشه اصلی شامل سه پوشه src ،models ، data است.

در پوشه data داده‌های اولیه مورد استفاده در کدها و بخش‌های مختلف قرار داده شده‌است. این داده‌ها عبارتند از داده‌های مربوط به توییت‌ها و تشخیص موضوعاتشان و همچنین مجموعه دادگان آموزشی و تست بخش تشخیص پروپاگاندا قرار دارد.

---

[7] Naïve Bayes
[8] Baseline
[9] Deep Neural Network
[10] Word Embedding
[11] BERT (Bidirectional Encoder Representations from Transformers)

در پوشه models فایل‌های ذخیره‌شده از مدل‌های بخش‌های مختلف بارگذاری شده‌است. و در پوشه src سورس کدهای سه بخش نامبرده قراره داده شده است.

کد نهایی من با نام test-rl در همین پوشه قرار داده شده است که از سایر ماژول‌های نوشته شده نیز استفاده می‌کند.

6 آزمایش‌ها

در این‌بخش به آزمون و خطاهای انجام شده در بخش تشخیص پروپاگاندا و هم چنین بخش آر ال پرداخته خواهد شد:

در بخش تشخیص پروپاگاندا ساختار شبکه به شرح زیر است:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 128)         3200000
_________________________________________________________________
bidirectional (Bidirectional (None, None, 256)         263168
_________________________________________________________________
bidirectional_1 (Bidirection (None, 256)               394240
_________________________________________________________________
dropout (Dropout)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 256)               65792
_________________________________________________________________
dense_1 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,956,225
Trainable params: 3,956,225
Non-trainable params: 0
```

تصویر 1: ساختار شبکه اولیه

همچنین نتایج زیر حاصل اجرای کد روی ۸ اپوک و با سایز دسته[12] ۶۴ می باشد:

[12] Batch

```
Epoch 1/8
235/235 [==============================] - ETA: 0s - loss: 0.5417 - accuracy: 0.7369 - f1: nan - recall: 0.1136 - precision: 0.3642
Epoch 00001: val_accuracy improved from -inf to 0.70652, saving model to LSTM_baseline.hdf5
235/235 [==============================] - 335s 1s/step - loss: 0.5417 - accuracy: 0.7369 - f1: nan - recall: 0.1136 - precision: 0.3642 - val_loss
Epoch 2/8
235/235 [==============================] - ETA: 0s - loss: 0.3979 - accuracy: 0.8262 - f1: 0.6319 - recall: 0.5795 - precision: 0.7287
Epoch 00002: val_accuracy did not improve from 0.70652
235/235 [==============================] - 334s 1s/step - loss: 0.3979 - accuracy: 0.8262 - f1: 0.6319 - recall: 0.5795 - precision: 0.7287 - val_l
Epoch 3/8
235/235 [==============================] - ETA: 0s - loss: 0.2517 - accuracy: 0.9011 - f1: 0.7997 - recall: 0.7713 - precision: 0.8462
Epoch 00003: val_accuracy did not improve from 0.70652
235/235 [==============================] - 327s 1s/step - loss: 0.2517 - accuracy: 0.9011 - f1: 0.7997 - recall: 0.7713 - precision: 0.8462 - val_l
Epoch 4/8
235/235 [==============================] - ETA: 0s - loss: 0.1403 - accuracy: 0.9506 - f1: 0.9025 - recall: 0.8872 - precision: 0.9246
Epoch 00004: val_accuracy did not improve from 0.70652
235/235 [==============================] - 332s 1s/step - loss: 0.1403 - accuracy: 0.9506 - f1: 0.9025 - recall: 0.8872 - precision: 0.9246 - val_l
Epoch 5/8
235/235 [==============================] - ETA: 0s - loss: 0.0979 - accuracy: 0.9647 - f1: 0.9310 - recall: 0.9160 - precision: 0.9516
Epoch 00005: val_accuracy did not improve from 0.70652
235/235 [==============================] - 326s 1s/step - loss: 0.0979 - accuracy: 0.9647 - f1: 0.9310 - recall: 0.9160 - precision: 0.9516 - val_l
Epoch 6/8
235/235 [==============================] - ETA: 0s - loss: 0.0739 - accuracy: 0.9731 - f1: 0.9470 - recall: 0.9338 - precision: 0.9646
Epoch 00006: val_accuracy did not improve from 0.70652
235/235 [==============================] - 326s 1s/step - loss: 0.0739 - accuracy: 0.9731 - f1: 0.9470 - recall: 0.9338 - precision: 0.9646 - val_l
Epoch 7/8
235/235 [==============================] - ETA: 0s - loss: 0.0566 - accuracy: 0.9796 - f1: 0.9605 - recall: 0.9524 - precision: 0.9712
Epoch 00007: val_accuracy did not improve from 0.70652
235/235 [==============================] - 327s 1s/step - loss: 0.0566 - accuracy: 0.9796 - f1: 0.9605 - recall: 0.9524 - precision: 0.9712 - val_l
Epoch 8/8
235/235 [==============================] - ETA: 0s - loss: 0.0525 - accuracy: 0.9787 - f1: 0.9586 - recall: 0.9494 - precision: 0.9707
Epoch 00008: val_accuracy did not improve from 0.70652
235/235 [==============================] - 328s 1s/step - loss: 0.0525 - accuracy: 0.9787 - f1: 0.9586 - recall: 0.9494 - precision: 0.9707 - val_l
```

تصویر 2: نتایج شبکه اولیه

Train accuracy: 0.9787

Train loss: 0.0525

Test accuracy: 0.6641

Test loss: 1.6621

همانطور که ملاحظه می کنید برای بهتر شدن دقت بر روی دادگان تست نیاز به تعویض پارامتر ها می باشد.

آزمایش شماره دو: بدون ***drop out***

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 128)         3200000
_________________________________________________________________
bidirectional (Bidirectional (None, None, 256)         263168
_________________________________________________________________
bidirectional_1 (Bidirection (None, 256)               394240
_________________________________________________________________
dense (Dense)                (None, 256)               65792
_________________________________________________________________
dense_1 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,956,225
Trainable params: 3,956,225
Non-trainable params: 0
```

تصویر 3: ساختار شبکه ثانویه

```
Epoch 1/8
235/235 [==============================] - ETA: 0s - loss: 0.5343 - accuracy: 0.7353 - f1: nan - recall: 0.1420 - precision: 0.3365
Epoch 00001: val_accuracy improved from -inf to 0.65332, saving model to LSTM_baseline.hdf5
235/235 [==============================] - 310s 1s/step - loss: 0.5343 - accuracy: 0.7353 - f1: nan - recall: 0.1420 - precision: 0.3365 - val_loss:
Epoch 2/8
235/235 [==============================] - ETA: 0s - loss: 0.3675 - accuracy: 0.8399 - f1: nan - recall: 0.6064 - precision: 0.7403
Epoch 00002: val_accuracy improved from 0.65332 to 0.70950, saving model to LSTM_baseline.hdf5
235/235 [==============================] - 308s 1s/step - loss: 0.3675 - accuracy: 0.8399 - f1: nan - recall: 0.6064 - precision: 0.7403 - val_loss:
Epoch 3/8
235/235 [==============================] - ETA: 0s - loss: 0.2040 - accuracy: 0.9211 - f1: 0.8443 - recall: 0.8293 - precision: 0.8730
Epoch 00003: val_accuracy did not improve from 0.70950
235/235 [==============================] - 307s 1s/step - loss: 0.2040 - accuracy: 0.9211 - f1: 0.8443 - recall: 0.8293 - precision: 0.8730 - val_los
Epoch 4/8
235/235 [==============================] - ETA: 0s - loss: 0.1062 - accuracy: 0.9600 - f1: 0.9214 - recall: 0.9074 - precision: 0.9412
Epoch 00004: val_accuracy did not improve from 0.70950
235/235 [==============================] - 308s 1s/step - loss: 0.1062 - accuracy: 0.9600 - f1: 0.9214 - recall: 0.9074 - precision: 0.9412 - val_los
Epoch 5/8
235/235 [==============================] - ETA: 0s - loss: 0.0680 - accuracy: 0.9749 - f1: 0.9496 - recall: 0.9392 - precision: 0.9639
Epoch 00005: val_accuracy did not improve from 0.70950
235/235 [==============================] - 308s 1s/step - loss: 0.0680 - accuracy: 0.9749 - f1: 0.9496 - recall: 0.9392 - precision: 0.9639 - val_los
Epoch 6/8
235/235 [==============================] - ETA: 0s - loss: 0.0490 - accuracy: 0.9820 - f1: 0.9641 - recall: 0.9531 - precision: 0.9777
Epoch 00006: val_accuracy did not improve from 0.70950
235/235 [==============================] - 308s 1s/step - loss: 0.0490 - accuracy: 0.9820 - f1: 0.9641 - recall: 0.9531 - precision: 0.9777 - val_los
Epoch 7/8
235/235 [==============================] - ETA: 0s - loss: 0.0399 - accuracy: 0.9840 - f1: 0.9684 - recall: 0.9606 - precision: 0.9787
Epoch 00007: val_accuracy did not improve from 0.70950
235/235 [==============================] - 308s 1s/step - loss: 0.0399 - accuracy: 0.9840 - f1: 0.9684 - recall: 0.9606 - precision: 0.9787 - val_los
Epoch 8/8
235/235 [==============================] - ETA: 0s - loss: 0.0377 - accuracy: 0.9843 - f1: 0.9691 - recall: 0.9603 - precision: 0.9800
Epoch 00008: val_accuracy did not improve from 0.70950
235/235 [==============================] - 309s 1s/step - loss: 0.0377 - accuracy: 0.9843 - f1: 0.9691 - recall: 0.9603 - precision: 0.9800 - val_los
```

تصویر 4: نتایج شبکه ثانویه

Train accuracy: 0.9843

Train loss: 0.0377

Test accuracy: 0.6718

Test loss: 1.8106

با این تغییر هم نتیجه تغییر خاصی نکرد.

بنابراین تصمیم به افزایش تعداد epochها با همان ساختار drop out دار گرفتم: (آزمایش شماره سه)

```
235/235 [==============================] - ETA: 0s - loss: 0.0493 - accuracy: 0.9799 - f1: 0.9614 - recall: 0.9529 - precision: 0.9728
Epoch 00008: val_accuracy did not improve from 0.71488
235/235 [==============================] - 324s 1s/step - loss: 0.0493 - accuracy: 0.9799 - f1: 0.9614 - recall: 0.9529 - precision: 0.9728 - val_l
Epoch 9/16
235/235 [==============================] - ETA: 0s - loss: 0.0479 - accuracy: 0.9833 - f1: 0.9672 - recall: 0.9591 - precision: 0.9779
Epoch 00009: val_accuracy did not improve from 0.71488
235/235 [==============================] - 325s 1s/step - loss: 0.0479 - accuracy: 0.9833 - f1: 0.9672 - recall: 0.9591 - precision: 0.9779 - val_l
Epoch 10/16
235/235 [==============================] - ETA: 0s - loss: 0.0404 - accuracy: 0.9843 - f1: 0.9691 - recall: 0.9583 - precision: 0.9824
Epoch 00010: val_accuracy did not improve from 0.71488
235/235 [==============================] - 325s 1s/step - loss: 0.0404 - accuracy: 0.9843 - f1: 0.9691 - recall: 0.9583 - precision: 0.9824 - val_l
Epoch 11/16
235/235 [==============================] - ETA: 0s - loss: 0.0425 - accuracy: 0.9834 - f1: 0.9675 - recall: 0.9587 - precision: 0.9786
Epoch 00011: val_accuracy did not improve from 0.71488
235/235 [==============================] - 324s 1s/step - loss: 0.0425 - accuracy: 0.9834 - f1: 0.9675 - recall: 0.9587 - precision: 0.9786 - val_l
Epoch 12/16
235/235 [==============================] - ETA: 0s - loss: 0.0303 - accuracy: 0.9876 - f1: 0.9760 - recall: 0.9714 - precision: 0.9818
Epoch 00012: val_accuracy did not improve from 0.71488
235/235 [==============================] - 324s 1s/step - loss: 0.0303 - accuracy: 0.9876 - f1: 0.9760 - recall: 0.9714 - precision: 0.9818 - val_l
Epoch 13/16
235/235 [==============================] - ETA: 0s - loss: 0.0284 - accuracy: 0.9880 - f1: 0.9762 - recall: 0.9700 - precision: 0.9842
Epoch 00013: val_accuracy did not improve from 0.71488
235/235 [==============================] - 325s 1s/step - loss: 0.0284 - accuracy: 0.9880 - f1: 0.9762 - recall: 0.9700 - precision: 0.9842 - val_l
Epoch 14/16
235/235 [==============================] - ETA: 0s - loss: 0.0296 - accuracy: 0.9886 - f1: 0.9776 - recall: 0.9737 - precision: 0.9829
Epoch 00014: val_accuracy did not improve from 0.71488
235/235 [==============================] - 325s 1s/step - loss: 0.0296 - accuracy: 0.9886 - f1: 0.9776 - recall: 0.9737 - precision: 0.9829 - val_l
Epoch 15/16
235/235 [==============================] - ETA: 0s - loss: 0.0243 - accuracy: 0.9899 - f1: 0.9804 - recall: 0.9737 - precision: 0.9885
Epoch 00015: val_accuracy did not improve from 0.71488
235/235 [==============================] - 329s 1s/step - loss: 0.0243 - accuracy: 0.9899 - f1: 0.9804 - recall: 0.9737 - precision: 0.9885 - val_l
Epoch 16/16
235/235 [==============================] - ETA: 0s - loss: 0.0246 - accuracy: 0.9891 - f1: 0.9790 - recall: 0.9791 - precision: 0.9800
Epoch 00016: val_accuracy did not improve from 0.71488
235/235 [==============================] - 325s 1s/step - loss: 0.0246 - accuracy: 0.9891 - f1: 0.9790 - recall: 0.9791 - precision: 0.9800 - val_l
```

تصویر 5: نتایج شبکه نهایی

Train accuracy: 0.9891

Train loss: 0.024

Test accuracy: 0.6748

Test loss: 2.4683

پس از مشاهده این نتایج نیز دریافتم که با این مدل لایه‌ای LSTM به نهایت دقتی که می‌توان روی داده تست رسید همان ۷۰ درصد است. از آنجا که دنبال درصد مطلوب‌تری برای کارهای آینده هستم، با بررسی مقالات مشابه ([5] و [6]) به این نتیجه رسیدم که ساختار برت برای این مدل تسک‌ها جواب خوبی خواهد داد.

7 مراحل و تحلیل:

همان‌طور که در ساختار پروژه اشاره شد، کد اصلی در فایل test_rl قرار داده شده است. در این بخش ابتدا دو ماژول subject detector و propaganda detector را بارگذاری می کنیم، زیرا در ادامه از آن‌ها استفاده خواهیم کرد. برای تست کردن بخش یادگیری تقویتی و اجرای پروژه به طور مثال از مجموعه توییت‌های اخیر دونالد ترامپ شروع کرده، با استفاده از ماژول تشخیص دهنده موضوعات، موضوع تک تک توییت‌ها را مشخص کرده و با تگی در کنارشان اضافه میکنیم. در نتیجه توییت‌هایمان بر حسب موضوع دسته بندی شده‌اند.

یکی از الگوریتم‌های معروف در زمینه‌ی مسائل N-arm bandit نمونه برداری تامسون می باشد که در این‌پروژه از آن استفاده کردم. خلاصه‌ای از این الگوریتم در تصویر زیر آورده شده‌است:

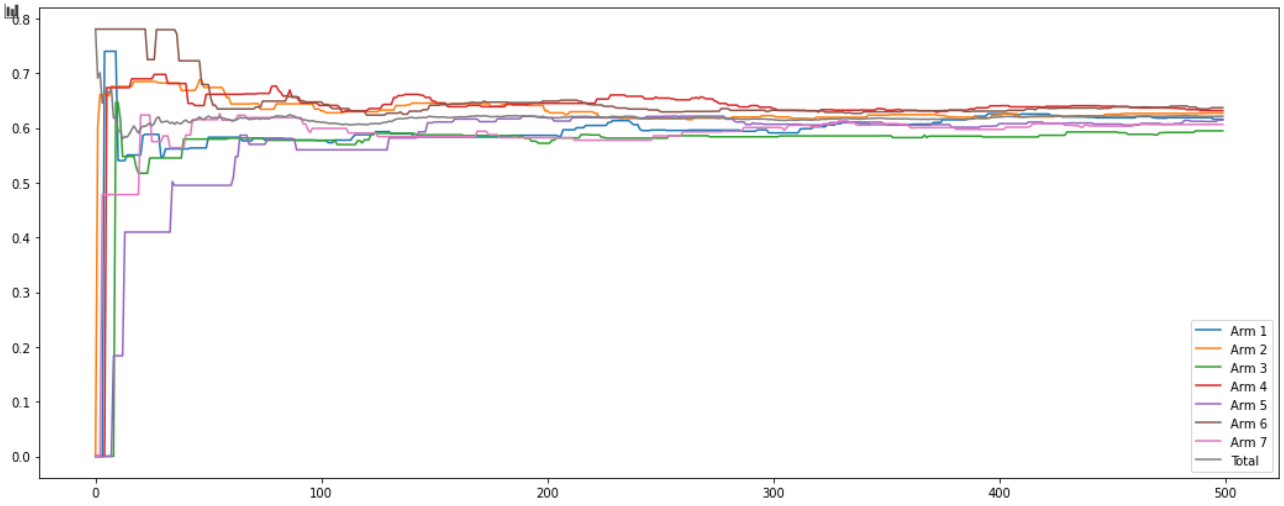
Input: $X, K = |X|$
$t \leftarrow 1$
$\forall x \in X : Success_x \leftarrow 0$
$\forall x \in X : Fails_x \leftarrow 0$
while $t \leq T$ **do**
  $\forall x \in X : \Theta_{x,t} \sim Beta(Success_x + 1, Fails_x + 1)$
  $x_t \leftarrow argmax(\Theta_{x,t})$
  play arm x_t and acquire u_t
  $Success_{x_t} \leftarrow Success_{x_t} + u_t$
  $Fails_{x_t} \leftarrow Fails_{x_t} + (1 - u_t)$

تصویر 6: خلاصه‌ای از الگوریتم نمونه‌برداری تامپسون

از توزیع بتا در این روش استفاده شده است. در حلقه‌ی اصلی برنامه در ابتدا یک عدد رندم و یک بازوی رندم شروع به کار کرده و توییتی در آن دسته (شماره بازو) انتخاب شده و توسط ماژول تشخیص پروپاگاندا پاداشی به آن نسبت داده شده است. (خروجی ماژول تشخیص پروپاگاندا احتمال دارا بودن یا نبودن پروپاگاندا در توییت را مشخص میکند)

حال طبق فرمول نمونه برداری تامسون به a و b مدنظر در آن دسته، r و 1-r را اضافه می‌کند. این اعمال در حلقه تکرار شده و نهایتا با بیرون آمدن از حلقه بیشترین مقدار پاداش میانگین (علت استفاده از میانگین این است که شاید فراوانی موردی کمتر باشد یا کمتر دیده شده باشد) به عنوان زمینه‌ای که آن شخص بیشتر از همه در آن تولید پروپاگاندا می کند، شناسایی می‌شود.

در رابطه با تعداد آزمایش‌ها[۱۳] و نقطه‌ی همگرایی[۱۴] نیز با توجه به نمودار بازوهای مختلف و آزمون و خطا به نتیجه رسیدم.



تصویر 7: نمودار نقطه همگرایی برحسب تعداد آزمایش‌ها برای بازوهای مختلف

7.1 آشنایی با نمونه برداری تامپسون[15]

نمونه برداری تامپسون به نام ویلیام آر تامپسون[۱۶]، برای انتخاب اقداماتی که معضل اکتشاف - بهره برداری در مسئله n-arm bandit را برطرف می کند ،یک ابتکار عمل است. این روش شامل انتخاب عملی است که حداکثر پاداش مورد انتظار را با توجه به یک باور تصادفی ترسیم می کند.

نمونه گیری تامپسون الگوریتمی برای مشکلات تصمیم گیری آنلاین است که در آن اقدامات به ترتیب انجام می شود که باید بین بهره برداری از آنچه برای به حداکثر رساندن عملکرد فوری و سرمایه گذاری برای جمع آوری اطلاعات جدید که

[13] Trials
[14] Convergence
[15] Thompson Sampling
[16] William R. Thompson

ممکن است عملکرد آینده را بهبود بخشد ، تعادل برقرار کند. این الگوریتم طیف گسترده ای از مشکلات را به روشی محاسباتی و کارآمد برطرف می کند و بنابراین از استفاده گسترده ای برخوردار است.[7]

8 مراجع

[1] wikipedia, "Propaganda," 28 Nov 2020. [Online]. Available: https://en.wikipedia.org/wiki/Propaganda.

[2] Wikipedia, "Multi-armed bandit," Multi-armed bandit, [Online]. Available: https://en.wikipedia.org/wiki/Multi-armed_bandit. [Accessed 22 october 2020].

[3] G. &. B.-C. A. &. W. H. &. P. R. &. N. P. Martino, "SemEval-2020 Task 11: Detection of Propaganda Techniques in News Articles.," 2020.

[4] A. Mishra, "Free Twitter Datasets Mega Compilation," 14 August 2019. [Online]. Available: https://www.trackmyhashtag.com/blog/twitter-datasets-free/.

[5] G.-A. a. T. M.-A. a. O. C. a. C. D.-C. Vlad, "Sentence-Level Propaganda Detection in News Articles with Transfer Learning and {BERT}-{B}i{LSTM}-Capsule Model," in *Association for Computational Linguistics*, Hong Kong, China, 2019.

[6] K. a. S. A. Aggarwal, "{NSIT}@{NLP}4{IF}-2019: Propaganda Detection from News Articles using Transfer Learning," in *Association for Computational Linguistics*, Hong Kong, China, 2019.