# گزارش تمرین چهارم شبکه عصبی

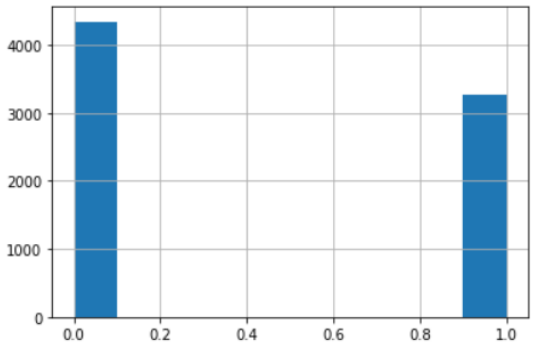
**علیرضا آزادبخت ۹۹۴۲۲۰۱۹**

در این دیتا ستی شامل ۷۶۱۳ توییت در رابطه با اتفاقات مصیبت بار طبیعی یا غیر طبیعی مانند زلزله و اتش سوزی و ... در اختیار ما قرار گرفته است و ۳۲۶۳ داده تست برای سابمیشن نیز به ما داده شده است. و هدف ما پیشبینی این است که آیا در یک توییت در مورد یکی از این اتفاقات صحبت شده یا موضوع دیگری در توییت بوده و یک توییت معمولی به حساب میآمده.

اولین ایده هایی که به ذهن میرسد این است که احتمالا بعضی از این کلمات معنای مصیبت بار دارند و بتوان با پیدا کردن آن ها این مسئله را حل کرد در ادامه ابتدا مراحل جست و جو در داده ها را انجام می‌دهیم و سپس به پاکسازی میپردازیم سپس به کمک مدل ترنسفورمر و مکانیزم attention به مدلینگ این مسئله میپردازیم.

## جست و جو داده:

ابتدا فراوانی کلاس های هدف را بررسی میکنیم:

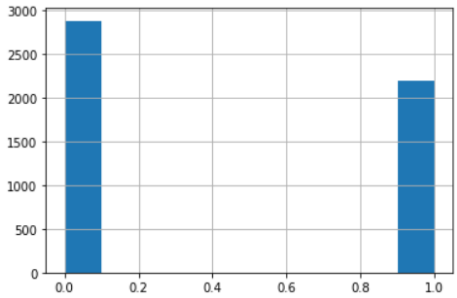



میتوان گفت که تقریبا کلاس ها برابر هستند میتوان بدون هیچ تغییری به مدل سازی پرداخت

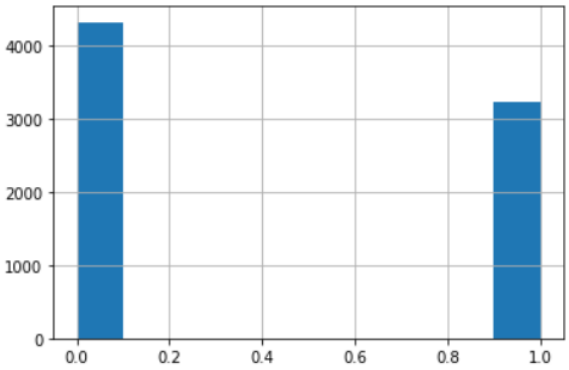
در مرحله بعدی بررسی میکنیم که از باقی فیچر های داده که لزوما پر هم نیستند و مقادیر نال دارند چه درصدی دارای دیتا هستند که از این بین فیچر location ۶۶ درصد داده keyword ۹۹ درصد داده و هر دوی این دو فیچر با هم هم ۶۶ درصد داده داشتند.

حال فراوانی کلاس هدف را برای داده هایی که این فیچر های آن ها نال نیست را بررسی میکنیم:

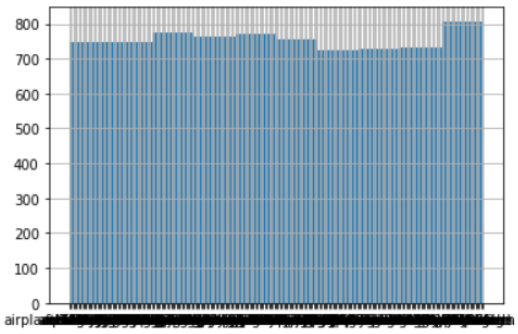
فراوانی داده هایی که لوکیشن آن ها نال نیست:




فراوانی کلاس های هدف که فیچر کی ورد آن ها نال نیست:




و در هر دو حالت فراوانی اصلی را مشاهده میکنیم، در این داده ها ۲۲۲ مقدار یکتا برای کی ورد و ۳۳۴۲ مقدار یکتا برای لوکیشن داریم، و توزیع کی ورد ها به صورت زیر میباشد:




نکته ای که از بررسی ها ه دست آوردیم این است که مقادیر قالب نداریم و تغریبا همه فیچر ها به صورت یکسانی اثر گذار هستند.

**پاکسازی و پیش پردازش توییت ها:**

در ابتدا به کمک یک مجموعه اماده از سر واژه ها و مخفف های فعل هایی نظیر he’sبه he is و w/e به whatever و یک سری ساده سازی های روتین کلمات در چت ها و توییت ها شروع به پک سازی کردیم و این کلمات را به فرم اصلی برگرداندیم

```
def clean_abbreviation(text):
    text = re.sub(r"he's", "he is", text)
    text = re.sub(r"there's", "there is", text)
    text = re.sub(r"We're", "We are", text)
    text = re.sub(r"That's", "That is", text)
    text = re.sub(r"won't", "will not", text)
    text = re.sub(r"they're", "they are", text)
    text = re.sub(r"Can't", "Cannot", text)
    text = re.sub(r"wasn't", "was not", text)
    text = re.sub(r"aren't", "are not", text)
    text = re.sub(r"isn't", "is not", text)
    text = re.sub(r"What's", "What is", text)
    text = re.sub(r"i'd", "I would", text)
    text = re.sub(r"should've", "should have", text)
    text = re.sub(r"where's", "where is", text)
    text = re.sub(r"we'd", "we would", text)
    text = re.sub(r"i'll", "I will", text)
    text = re.sub(r"weren't", "were not", text)
    text = re.sub(r"They're", "They are", text)
    text = re.sub(r"let's", "let us", text)
    text = re.sub(r"it's", "it is", text)
    text = re.sub(r"can't", "cannot", text)
    text = re.sub(r"don't", "do not", text)
    text = re.sub(r"you're", "you are", text)
    text = re.sub(r"i've", "I have", text)
    text = re.sub(r"that's", "that is", text)
    text = re.sub(r"i'll", "I will", text)
    text = re.sub(r"doesn't", "does not", text)
    text = re.sub(r"i'd", "I would", text)
    text = re.sub(r"didn't", "did not", text)
    text = re.sub(r"ain't", "am not", text)
    text = re.sub(r"you'll", "you will", text)
    text = re.sub(r"I've", "I have", text)
    text = re.sub(r"Don't", "do not", text)
    text = re.sub(r"I'll", "I will", text)
    text = re.sub(r"I'd", "I would", text)
    text = re.sub(r"Let's", "Let us", text)
    text = re.sub(r"you'd", "You would", text)
    text = re.sub(r"It's", "It is", text)
    text = re.sub(r"Ain't", "am not", text)
    text = re.sub(r"Haven't", "Have not", text)
    text = re.sub(r"Could've", "Could have", text)
    text = re.sub(r"youve", "you have", text)
    text = re.sub(r"haven't", "have not", text)
    text = re.sub(r"hasn't", "has not", text)
    text = re.sub(r"There's", "There is", text)
    text = re.sub(r"He's", "He is", text)
    text = re.sub(r"It's", "It is", text)
    text = re.sub(r"You're", "You are", text)
    text = re.sub(r"I'M", "I am", text)
    text = re.sub(r"shouldn't", "should not", text)
    text = re.sub(r"wouldn't", "would not", text)
    text = re.sub(r"i'm", "I am", text)
    text = re.sub(r"I'm", "I am", text)
    text = re.sub(r"Isn't", "is not", text)
    text = re.sub(r"Here's", "Here is", text)
    text = re.sub(r"you've", "you have", text)
    text = re.sub(r"we're", "we are", text)
    text = re.sub(r"what's", "what is", text)
    text = re.sub(r"couldn't", "could not", text)
    text = re.sub(r"we've", "we have", text)
    text = re.sub(r"who's", "who is", text)
    text = re.sub(r"y'all", "you all", text)
    text = re.sub(r"would've", "would have", text)
    text = re.sub(r"it'll", "it will", text)
```

```
    text = re.sub(r"we'll","we will", text)
    text = re.sub(r"We've", "We have", text)
    text = re.sub(r"he'll","he will", text)
    text = re.sub(r"Y'all", "You all", text)
    text = re.sub(r"Weren't", "Were not", text)
    text = re.sub(r"Didn't", "Did not", text)
    text = re.sub(r"they'll", "they will", text)
    text = re.sub(r"they'd", "they would", text)
    text = re.sub(r"DON'T", "DO NOT", text)
    text = re.sub(r"they've", "they have", text)

    text = re.sub(r"tnwx", "Tennessee Weather", text)
    text = re.sub(r"azwx", "Arizona Weather", text)
    text = re.sub(r"alwx", "Alabama Weather", text)
    text = re.sub(r"wordpressdotcom","wordpress", text)
    text = re.sub(r"gawx", "Georgia Weather", text)
    text = re.sub(r"scwx","South Carolina Weather", text)
    text = re.sub(r"cawx", "California Weather", text)
    text = re.sub(r"usNWSgov", "United States National Weather Service", text)
    text = re.sub(r"MH370", "Malaysia Airlines Flight 370", text)
    text = re.sub(r"okwx", "Oklahoma City Weather", text)
    text = re.sub(r"arwx", "Arkansas Weather", text)
    text = re.sub(r"lmao","laughing my ass off", text)
    text = re.sub(r"amirite", "am I right", text)

    text = re.sub(r"w/e", "whatever", text)
    text = re.sub(r"w/", "with", text)
    text = re.sub(r"USAgov", "USA government", text)
    text = re.sub(r"recentlu","recently", text)
    text = re.sub(r"Ph0tos", "Photos", text)
    text = re.sub(r"exp0sed", "exposed", text)
    text = re.sub(r"<3", "love", text)
    text = re.sub(r"amageddon", "armageddon", text)
    text = re.sub(r"Trfc", "Traffic", text)
    text = re.sub(r"WindStorm","Wind Storm", text)
    text = re.sub(r"16yr", "16 year", text)
    text = re.sub(r"TRAUMATISED", "traumatized", text)

    text = re.sub(r"IranDeal", "Iran Deal", text)
    text = re.sub(r"ArianaGrande", "Ariana Grande", text)
    text = re.sub(r"camilacabello97", "camila cabello", text)
    text = re.sub(r"RondaRousey","Ronda Rousey", text)
    text = re.sub(r"MTVHottest","MTV Hottest", text)
    text = re.sub(r"TrapMusic", "Trap Music", text)
    text = re.sub(r"ProphetMuhammad", "Prophet Muhammad", text)
    text = re.sub(r"PantherAttack", "Panther Attack", text)
    text = re.sub(r"StrategicPatience", "Strategic Patience", text)
    text = re.sub(r"socialnews","social news", text)
    text = re.sub(r"IDPs:", "Internally Displaced People :", text)
    text = re.sub(r"ArtistsUnited", "Artists United", text)
    text = re.sub(r"ClaytonBryant", "Clayton Bryant", text)
    text = re.sub(r"jimmyfallon", "jimmy fallon", text)
    text = re.sub(r"justinbieber", "justin bieber", text)
    text = re.sub(r"Time2015", "Time 2015", text)
    text = re.sub(r"djicemoon", "dj icemoon", text)
    text = re.sub(r"LivingSafely", "Living Safely", text)
    text = re.sub(r"FIFA16", "Fifa 2016", text)
    text = re.sub(r"thisiswhywecanthavenicethings", "this is why we cannot have nice things", text)
    text = re.sub(r"bbcnews", "bbc news", text)
    text = re.sub(r"UndergroundRailraod","Underground Railraod", text)
    text = re.sub(r"c4news","c4 news", text)
    text = re.sub(r"MUDSLIDE", "mudslide", text)
    text = re.sub(r"NoSurrender", "No Surrender", text)
    text = re.sub(r"NotExplained", "Not Explained", text)
    text = re.sub(r"greatbritishbakeoff", "great british bake off", text)
    text = re.sub(r"LondonFire", "London Fire", text)
    text = re.sub(r"KOTAWeather", "KOTA Weather", text)
    text = re.sub(r"LuchaUnderground", "Lucha Underground", text)
    text = re.sub(r"KOIN6News", "KOIN 6 News", text)
    text = re.sub(r"LiveOnK2", "Live On K2", text)
    text = re.sub(r"9NewsGoldCoast", "9 News Gold Coast", text)
    text = re.sub(r"nikeplus", "nike plus", text)
    text = re.sub(r"david_cameron", "David Cameron", text)
```

```
text = re.sub(r"peterjukes", "Peter Jukes", text)
text = re.sub(r"MikeParrActor", "Michael Parr", text)
text = re.sub(r"4PlayThursdays", "Foreplay Thursdays", text)
text = re.sub(r"TGF2015", "Tontitown Grape Festival", text)
text = re.sub(r"realmandyrain", "Mandy Rain", text)
text = re.sub(r"GraysonDolan", "Grayson Dolan", text)
text = re.sub(r"ApolloBrown", "Apollo Brown", text)
text = re.sub(r"saddlebrooke", "Saddlebrooke", text)
text = re.sub(r"TontitownGrape", "Tontitown Grape", text)
text = re.sub(r"AbbsWinston", "Abbs Winston", text)
text = re.sub(r"ShaunKing", "Shaun King", text)
text = re.sub(r"MeekMill", "Meek Mill", text)
text = re.sub(r"TornadoGiveaway", "Tornado Giveaway", text)
text = re.sub(r"GRupdates", "GR updates", text)
text = re.sub(r"SouthDowns", "South Downs", text)
text = re.sub(r"braininjury", "brain injury", text)
text = re.sub(r"auspol", "Australian politics", text)
text = re.sub(r"PlannedParenthood", "Planned Parenthood", text)
text = re.sub(r"calgaryweather", "Calgary Weather", text)
text = re.sub(r"weallheartonedirection", "we all heart one direction", text)
text = re.sub(r"edsheeran", "Ed Sheeran", text)
text = re.sub(r"TrueHeroes", "True Heroes", text)
text = re.sub(r"ComplexMag", "Complex Magazine", text)
text = re.sub(r"TheAdvocateMag", "The Advocate Magazine", text)
text = re.sub(r"CityofCalgary", "City of Calgary", text)
text = re.sub(r"EbolaOutbreak", "Ebola Outbreak", text)
text = re.sub(r"SummerFate", "Summer Fate", text)
text = re.sub(r"RAmag", "Royal Academy Magazine", text)
text = re.sub(r"offers2go", "offers to go", text)
text = re.sub(r"ModiMinistry", "Modi Ministry", text)
text = re.sub(r"TAXIWAYS", "taxi ways", text)
text = re.sub(r"Calum5SOS", "Calum Hood", text)
text = re.sub(r"JamesMelville", "James Melville", text)
text = re.sub(r"JamaicaObserver", "Jamaica Observer", text)
text = re.sub(r" textLikeItsSeptember11th2001", " text like it is september 11th 2001", text)
text = re.sub(r"cbplawyers", "cbp lawyers", text)
text = re.sub(r"fewmore texts", "few more texts", text)
text = re.sub(r"BlackLivesMatter", "Black Lives Matter", text)
text = re.sub(r"NASAHurricane", "NASA Hurricane", text)
text = re.sub(r"onlinecommunities", "online communities", text)
text = re.sub(r"humanconsumption", "human consumption", text)
text = re.sub(r"Typhoon-Devastated", "Typhoon Devastated", text)
text = re.sub(r"Meat-Loving", "Meat Loving", text)
text = re.sub(r"facialabuse", "facial abuse", text)
text = re.sub(r"LakeCounty", "Lake County", text)
text = re.sub(r"BeingAuthor", "Being Author", text)
text = re.sub(r"withheavenly", "with heavenly", text)
text = re.sub(r"thankU", "thank you", text)
text = re.sub(r"iTunesMusic", "iTunes Music", text)
text = re.sub(r"OffensiveContent", "Offensive Content", text)
text = re.sub(r"WorstSummerJob", "Worst Summer Job", text)
text = re.sub(r"HarryBeCareful", "Harry Be Careful", text)
text = re.sub(r"NASASolarSystem", "NASA Solar System", text)
text = re.sub(r"animalrescue", "animal rescue", text)
text = re.sub(r"KurtSchlichter", "Kurt Schlichter", text)
text = re.sub(r"Throwingknifes", "Throwing knives", text)
text = re.sub(r"GodsLove", "God's Love", text)
text = re.sub(r"bookboost", "book boost", text)
text = re.sub(r"ibooklove", "I book love", text)
text = re.sub(r"NestleIndia", "Nestle India", text)
text = re.sub(r"realDonaldTrump", "Donald Trump", text)
text = re.sub(r"DavidVonderhaar", "David Vonderhaar", text)
text = re.sub(r"CecilTheLion", "Cecil The Lion", text)
text = re.sub(r"weathernetwork", "weather network", text)
text = re.sub(r"GOPDebate", "GOP Debate", text)
text = re.sub(r"RickPerry", "Rick Perry", text)
text = re.sub(r"frontpage", "front page", text)
text = re.sub(r"NewsIn texts", "News In texts", text)
text = re.sub(r"ViralSpell", "Viral Spell", text)
text = re.sub(r"til_now", "until now", text)
text = re.sub(r"volcanoinRussia", "volcano in Russia", text)
text = re.sub(r"ZippedNews", "Zipped News", text)
text = re.sub(r"MicheleBachman", "Michele Bachman", text)
```

```
    text = re.sub(r"53inch","53 inch", text)
    text = re.sub(r"KerrickTrial","Kerrick Trial", text)
    text = re.sub(r"abstorm","Alberta Storm", text)
    text = re.sub(r"Beyhive","Beyonce hive", text)
    text = re.sub(r"RockyFire","Rocky Fire", text)
    text = re.sub(r"Listen/Buy","Listen / Buy", text)
    text = re.sub(r"ArtistsUnited","Artists United", text)
    text = re.sub(r"ENGvAUS","England vs Australia", text)
    text = re.sub(r"ScottWalker","Scott Walker", text)
    return text
```

در مرحله بعدی تمامی حروف توییت ها را به حروف کوچک تبدیل میکنیم و استاپ ورد ها و کلماتی که معنایی خاصی ندارند و در زبان انگلیسی برای فهم بهتر انسانی استفاده میشوند اما برای مسئله طبقه بندی مفید نیستند به دلیل پرتکرار بودن از داده ها حذف میکنیم و علائم نگارشی مثل نقطه و علامت سوال و ... را نیز دور میریزیم و اگر کلمه ای در توییت ها باشد که در زبان انگلیسی نباشه آن را نیز حذف میکنیم مثل کلمات از زبان های دیگر و ایموجی ها و اعداد

در توییت ها بعضا دیده میشود که کلمات را کش دار مینویسند مثلا aaaand its gooone این کلمات را نیز در مرحله پاک سازی به فرم اصلی خود بر میگردانیم و به and its gone تبدیل میکنیم، برای اینکه ممکن از توییت هایی که دارای عدد هستند دارای اطلاعات خاصی باشند مثلا امار کشته شده ها یا شدت یک زلزله و.. باشد به جای همه اعداد یک توکن جدید به نام isnumber میگذاریم که اگر در این اعداد اطلاعات مفیدی بود مدل خودش آن ها را استفاده کند.

در مرحله بعد باید از بین دو روش stem و lemmaانتخاب کنیم این دو روش وظیفه یکسانی را دنبال میکنند اما روش اول سریعتر عمل میکند و روش دوم به یک مرحله تگ گذاری نوع کلمات از قبیل فعل اسم و صفت نیاز دارد تا بتواند به درستی ریشه کلمات را پیدا کند، در این مطالعه ما از روش lemma استفاده کردیم و به کمک pos_tag کتابخانه nltk نوع کلمات را شناسایی کردیم و سپس به کمک wordNetLemmatizerکلمات را به ریشه های اصلی آن ها تبدیل کردیم و با این کار به طور کلی توییت ها را یک دست و تمیز کردیمو آماده مدل سازی آن هستیم.

**مدل سازی:**

در ابتدا داده های آموزش را به دو کلاس تست و آموزش تقسیم کردیم با نسبت ۲۰ درصد تقسیم کردیم برای اینکه بتوانیم با ترنسفرمر ها این تسک را انجام دهیم نیاز داریم کلمات را به یک فضای برداری ببریم هر کلمه

را با یک بردار نمایش دهیم، برای این کار از مدل از پیش آموزش دیده bert-base-uncased استفاده کردیم و تمامی کلمات را به این فضا ببرداری بردیم سپس یک زیرو پدینگ با بیشترین طول ۶۴ در نظر گرفتیم که اگر کمتر از ۶۴ باشد به انتهای آن صفر اضافه میکنیم و اگر بیشتر باشد باقی آن را دور میریزیم بعد از این امبدیگی که نجام دادیم دده های ورودی به رشته هایی با طول ۶۴ و اندازه ۷۶۸ تبدیل شدند.

مدل ترنسفورمر خود را به صورت زیر میسازیم شامل یک ماژول انکدر ترنسفورمر با بردار های ورودی ۷۶۸ تا و ۳۲ هد attention بعد شبکه فولی کانکت ۲۵۶ تایی است که چهار بار تکرار میشود یعنی ۴ انکدر روی هم سوار هستند و در پایان یک لایه مکس پولینگ انجام میدهیم و سپس بردار ورودی را را فلت میکنیک و به یک شبکه فولی کانکت میدهیم که خروجی آن دو نورون است. معماری شبکه را در زیر مشاهده میکنیم.

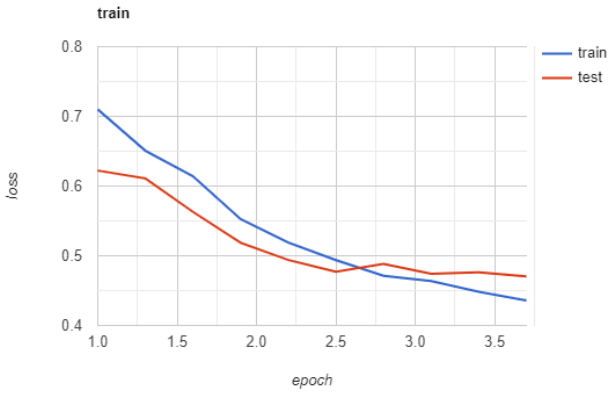
```
TransformerClassifier(
 (encoder): TransformerEncoder(
  (layers): ModuleList(
   (0): TransformerEncoderLayer(
    (self_attn): MultiheadAttention(
     (out_proj): _LinearWithBias(in_features=768, out_features=768, bias=True)
    )
    (linear1): Linear(in_features=768, out_features=256, bias=True)
    (dropout): Dropout(p=0.1, inplace=False)
    (linear2): Linear(in_features=256, out_features=768, bias=True)
    (norm1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (norm2): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (dropout1): Dropout(p=0.1, inplace=False)
    (dropout2): Dropout(p=0.1, inplace=False)
   )
   (1): TransformerEncoderLayer(
    (self_attn): MultiheadAttention(
     (out_proj): _LinearWithBias(in_features=768, out_features=768, bias=True)
    )
    (linear1): Linear(in_features=768, out_features=256, bias=True)
    (dropout): Dropout(p=0.1, inplace=False)
    (linear2): Linear(in_features=256, out_features=768, bias=True)
    (norm1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (norm2): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (dropout1): Dropout(p=0.1, inplace=False)
    (dropout2): Dropout(p=0.1, inplace=False)
   )
   (2): TransformerEncoderLayer(
    (self_attn): MultiheadAttention(
     (out_proj): _LinearWithBias(in_features=768, out_features=768, bias=True)
    )
    (linear1): Linear(in_features=768, out_features=256, bias=True)
    (dropout): Dropout(p=0.1, inplace=False)
    (linear2): Linear(in_features=256, out_features=768, bias=True)
    (norm1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (norm2): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
    (dropout1): Dropout(p=0.1, inplace=False)
    (dropout2): Dropout(p=0.1, inplace=False)
```

```
    )
    (3): TransformerEncoderLayer(
     (self_attn): MultiheadAttention(
       (out_proj): _LinearWithBias(in_features=768, out_features=768, bias=True)
     )
     (linear1): Linear(in_features=768, out_features=256, bias=True)
     (dropout): Dropout(p=0.1, inplace=False)
     (linear2): Linear(in_features=256, out_features=768, bias=True)
     (norm1): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
     (norm2): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
     (dropout1): Dropout(p=0.1, inplace=False)
     (dropout2): Dropout(p=0.1, inplace=False)
    )
   )
  )
  (flatten): Sequential(
   (0): MaxPool2d(kernel_size=(3, 3), stride=(3, 3), padding=0, dilation=1, ceil_mode=False)
   (1): Flatten(start_dim=1, end_dim=-1)
  )
  (linear): Linear(in_features=5376, out_features=2, bias=True)
)
```

شبکه فوق را به کمک بهینه ساز adam و تابع خطا crossEntropyLoss با و نرخ یادگیری ۰.۰۰۰۰۵ آموزش میدهیم

فرایند یادگیری به دلیل کم بودن تعداد ایپاک ها نسبتا سریع عمل میکند

روند یادگیری مدل:



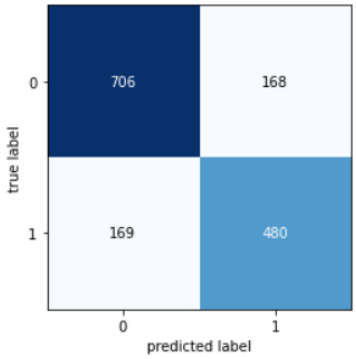

و در پایان آموزش خطا بر روی داده آموزش ۰.۴۴ و بر روی داده تست ۰.۴۷ میباشد.

به طور کلاسیک تر دقت مدل را بر روی مجموعه داده تست بدست میآوریم:

```
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.81      0.81       874
           1       0.74      0.74      0.74       649

    accuracy                           0.78      1523
   macro avg       0.77      0.77      0.77      1523
weighted avg       0.78      0.78      0.78      1523
```




**نتیجه :**

بعد از سابمیت کردن نتایج مدل به مسابقه کگل دقت 0.75697 را کسب کردیم که به دلیل درست نبودن لیدر برد نمیتوانیم دقت مدل را بسنجیم چون برچسب های نهایی مسابقه در اینترنت موجود میباشد و خیلی از ریزالت های کسب شده قابل اعتماد نیستند، اما دو جا امکان بهتر شدن داریم یک به دلیل مشکلات و محدودیت های سخت افزاری نتوانستیم طول رشته ورودی را بیشتر از ۶۴ بگیریم و تعداد زیادی از اطلاعات را از دست دادیم، دو از فیچر های دیگر با این پیشفرض که توزیع های آن ها یکسان هستند استفاده نکردیم و میشد خروجی شبکه را به عنوان یک فیچر کنار باقی فیجر ها قرار دهیم و به کمک یک مدل درختی به حل بپردازیم و دقت بهتری کسب کنیم اما به دلیل مشکلات گفته شده این ایده بررسی نشد. میتوانستیم به جای استفاده از مدل های از پیش آموزش داده شده برای امبدینگ از همراه شبکه استفاده کنیم و با آن همزمان آموزش داده شود اما چون خیلی این کار بهینه نبود و کرنل های کگل کرش میکردند انجام ندادیم.