УНИВЕРЗИТЕТ У НИШУ
ЕЛЕКТРОНСКИ ФАКУЛТЕТ

Методе за аугментацију текста

Истраживање различитих техника и њихов утицај на перформансе класификационих модела и дубоких неуронских мрежа

Пројекат
Предмет: Обрада природних језика

Студент:

Настасија Станковић, бр. инд. 1622

Ментор:

Сузана Стојковић

Ниш,2024. година

Садржај

Ниш,2024. година

1. Увод

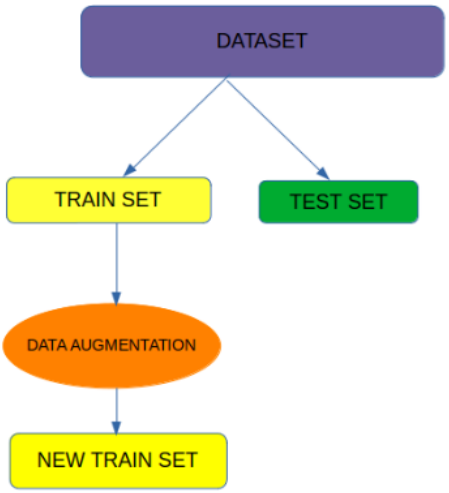
Аугментација текста је кључна техника у обради природног језика, са циљем побољшања перформанси модела за различите задатке као што су анализа сентимента, класификација тема и анализа намера. Потреба за аугментацијом текста произилази из честог проблема недостатка означених података у реалним апликацијама, што ограничава могућност развоја и примене ефикасних модела.

Недостатак означених података представља значајан изазов приликом развоја модела, посебно у доменима где је прикупљање означених података скупо, временски захтевно или недоступно. У таквим сценаријима, примена традиционалних метода дубоког учења може бити ограничена, а модели могу показати лошу генерализацију на новим, невиђеним подацима. Овај проблем је посебно изражен у дубоким неуронским мрежама које захтевају велике количине података за обуку.

Аугментација текста омогућава генерисање нових података из постојећих скупова података, чиме се проширује тренинг скуп и унапређује способност модела да генерализује на непознате податке. Ова техника се показала као кључна у побољшању перформанси модела у условима ограничених ресурса.

Осим што омогућава проширење тренинг скупа, аугментација текста такође помаже у избегавању оверфиттинга модела на постојећим подацима. Додавањем разноликости у тренинг скуп, модели постају отпорнији на шум у подацима и генерализују боље на нове, невиђене инстанце. Ово је посебно корисно за дубоке неуронске мреже, које су склоне оверфиттингу због велике капацитета за учење сложених образаца у подацима.

Кроз истраживање техника аугментације текста, имамо прилику да истражимо различите приступе генерисању нових података, као и да анализирамо њихов утицај на перформансе класификационих модела и дубоких неуронских мрежа. Разумевање и примена ових техника омогућава нам да развијемо ефикасне моделе за обраду текста који су отпорни на недостатак означених података и показују високу способност генерализације на различите домене и задатке.



Слика 1. Примена аугментације података

2. Припрема текста за аугментацију

Технике за аугментацију текста представљају технике којима се обрађују подаци који се у изворном облику налазе у текстуалном формату.

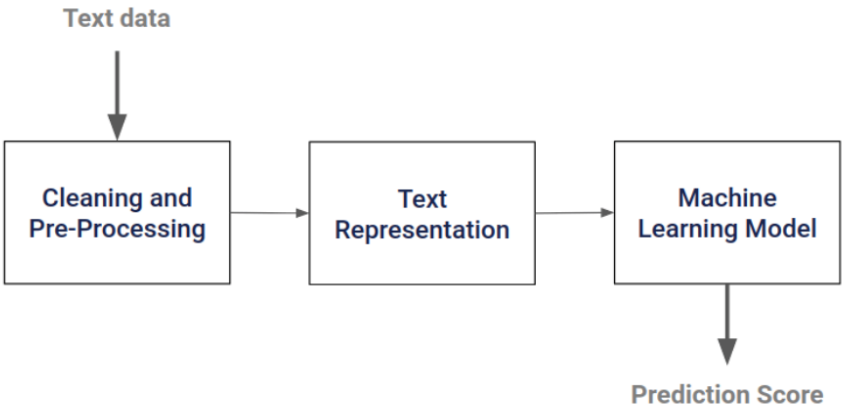
Обзиром да над подацима који се изворно налазе у текстуалном формату није могуће применити даље кораке (попут примене математичких метода машинског учења или тренирања модела), такве податке је најпре неопходно претворити у погодан формат који се може обрађивати. Погодан формат јесте векторски/тензорски простор у којем је неопходно представити текстуални контекст посматраног скупа података.

Обрађени текстуални подаци постају вектори који се у векторском/тензорском простору представљају као низови бројева одређене дужине. Вредности које ови вектори носе са собом треба да на високо-квалитативан начин опишу дати текстуални податак при чему треба одржати својства текстова попут редоследа речи у реченици, контекстуалне зависности између речи, дужина текстова, контекстуална сличности и сл.

2.1 Препроцесирање података

Препроцесирање података се бави основним чишћењем улазних текстуалних података и подразумева неке од следећи техника:

- Чишћење текста
- Нормализација
- Токенизација
- Лематизација и стемовање
- Уклањање стоп-речи



Слика 2. Процес припреме текста

2.1.1 Чишћење текста

Чишћење текста је први корак у препроцесирању текстуалних података и подразумева:

- **Уклањање непотребних знакова**: Идентификација и уклањање специјалних знакова, интерпункцијских знакова, емотикона и других непотребних симбола који не доприносе значењу текста.

- **Уклањање HTML ознака**: Ако се текст добија из web страница, често може садржавати HTML ознаке које нису релевантне за анализу текста. Ове ознаке треба уклонити.

- **Уклањање бројева**: Ако анализирамо текст који није нумерички, често је корисно уклонити бројеве како би се фокусирали само на лингвистичке карактеристике текста.

- **Уклањање непознатих знакова**: Идентификација и уклањање знакова који нису део стандардног скупа знакова језика који се анализира.

- **Уклањање непотребних размака**: Уклањање вишеструких узастопних размака и размака на почетку и крају текста.

2.1.2 Нормализација

Нормализација подразумева поступак претварања текста у стандардни формат или облик како би се олакшала даља анализа. Ево неколико аспеката нормализације текста:

- **Конверзија у велика или мала слова**: Нормализација може укључивати конверзију свих слова у тексту у велика или мала слова. Ово је корисно јер омогућава доследност у обради текста, независно од формата у којем је текст оригинално написан. На пример, "Добро јУтРо" би се могло конвертовати у "добро јутро" или "ДОБРО ЈУТРО", зависно од потреба анализе.

- **Претварање у стандардни формат**: Нормализација може укључивати претварање различитих варијација истог израза или речи у стандардни облик. На пример, може се претворити "кошарка" и "кошарке" у исти облик "кошарка".

- **Претварање у јединствени стандардни формат**: Нормализација такође може укључивати претварање различитих формата датума, бројева телефона, адреса и других ентитета у јединствени стандардни формат. На пример, све датуме можемо претворити у формат "ДД.ММ.ГГГГ".

- **Уклањање дупликата**: Нормализација такође може укључивати уклањање дупликата речи или реченица које се могу појавити у тексту, што олакшава даљу анализу.

2.1.3 Токенизација

Токенизација је основни корак у обради текстуалних података који се састоји у дељењу улазне реченице или текста на мање јединице, које називамо токенима. Ови токени могу бити појединачне речи, делови речи, чак и појединачна слова, у зависности од конкретних захтева.

Неколико кључних аспеката токенизације:

- **Дељење текста на токене**: Основни циљ токенизације је поделити текст на смислене јединице које чине основу за даљу анализу. Ове јединице могу бити речи, делови речи, интерпункцијски знакови или чак слова.

- **Представљање смислене целине**: Токени треба да представљају смислену целину која има одређени контекст или значење. На пример, реч "необичан" може бити један токен, док би реч "не" и "обичан" биле два одвојена токена.

- **Прилагођавање специфичним захтевима**: Токенизација се може прилагодити специфичним захтевима анализе текста. На пример, у неким случајевима можда желимо да сачувамо интерпункцијске знакове или да делимо текст на нивоу реченица уместо на нивоу речи.

- **Погодност за даљу обраду**: Након токенизације, текстуални подаци се обично представљају као листа токена, што олакшава даљу анализу. Ова листа токена може се даље користити у процесима као што су лематизација, уклањање стоп речи, векторизација и слично.

Примена токенизације пре техника за аугментацију омогућава нам већу контролу над процесом обраде текста, очување семантике текста и ефикаснију обраду података.

2.1.4 Лематизација и стемовање

Ове технике се користе за конверзију и скраћивање речи на њихов основни облик или корен речи. Технике се примењују на идентичан начин при чему се примењују различите трансформације.

Стемовање представља једноставнији процес обраде речи при чему се посматрана реч своди на корен речи изведен из посматране. Додатна битна карактеристика јесте и време извршавања самог процеса обраде података. Такође, стемовање се обавља над сваком речју појединачно без додатног познавања контекста осталих речи у околини.

Лематизација представља сложенији процес обраде текста у односу на стемизацију јер у процесу обраде података узима шири контекст и већи број речи како би креирао „лему“. Лема представља основну форму речи, тј. облик речи који представља њено лексично значење или основни облик. Лематизација је процес редукције речи при чему се све варијације одређене речи своде на идентичан лексички облик.

Ове технике помажу у оптимизацији процеса аугментације, смањењу дупликата и побољшању способности модела да генерализује информације из текста.

2.1.5 Уклањање стоп-речи

Стоп-речи представљају специфичан корпус речи сваког језика које са собом не носе високи значај у обради и анализи текста. Овакве речи је приликом обраде текста могуће избацити из скупа улазних реченица јер не доприносе квалитативним својствима посматраних текстова, док повећавају време процеса обраде текста. Ове речи се сматрају често коришћеним у језику и не доприносе контекстуалној репрезентацији улазних реченица.

Стоп-речи су углавном често коришћене речи, везници, предлози, речце и сл. Примери оваквих речи у енглеском речнику су следеће: “the”, “a”, “and”, “is”, “in”, “into”…

Циљ овог корака је смањење величине текста и времена потребног за обраду, али и фокусирање на речи које носе више значаја за анализу. Уклањањем стоп-речи, фокус се премешта на значајне речи које могу бити корисне за класификацију или анализу, што често доприноси бољим резултатима при обради текста.

Међутим, потребно је водити рачуна да се не уклоне речи које су неопходне за одржавање смисла и контекста у реченици.

3. Преглед техника за аугментацију текста

Технике које ће у овом делу бити обрађене су:

- Аугментација карактера
- Аугментација речи
- Аугментација реченица

3.1 Аугментација карактера

Свака техника аугментације карактера пружа могућност манипулације текстуалних података на нивоу карактера, отварајући пут ка стварању разноликих варијација истог текста. Ове технике су корисне за побољшање перформанси модела обраде природног језика, као и за повећање њихове робустности кроз већу разноликост података.

У наставку ћемо прегледати неколико кључних техника аугментације карактера.

3.1.1 OCR augmenter

OCR (*Оптичко препознавање карактера*) је кључна техника за побољшање квалитета текста препознатог са слика или скенираних докумената. Као део процеса OCR-а,генеришу се различите варијације текста које могу симулирати грешке и неправилности које се могу јавити током процеса препознавања текста са слика.

Кључне функције OCR augmentera:

- **Симулација грешака OCR-а**: OCR аугментер може симулирати уобичајене грешке које се јављају током процеса препознавања карактера са слика или скенираних докумената. Ово укључује грешке у препознавању појединачних карактера, као и изостављање или замену речи.

- **Додавање шума**: Имплементација шума у текст омогућава симулацију стварних услова скенирања докумената, укључујући неправилности у осветљењу или оштећења на документима. Ово помаже у побољшању отпорности система за препознавање текста на различите услове и варијације текста.

- **Замена или измена карактера**: OCR аугментер омогућава замену или измену одређених карактера у тексту, чиме се симулирају различите варијације карактера које се могу појавити током процеса препознавања текста са слика или скенираних докумената.

- **Промена фонта или стилизација текста**: Ова функција омогућава промену фонта или стилизацију текста на начин који може бити различит од оригиналног текста. Тиме се симулирају различити стилови и формати текста који се могу појавити на сликама или скенираним документима.

- **Генерисање варијација сличних речи**: OCR аугментор може генерисати варијације речи које су сличне оригиналним речима, али се могу разликовати у неким карактеристикама као што су дужина или сложеност карактера.

Дакле, када применимо OCR аугментор на одређени текст, резултат ће бити модификовани текст са различитим варијацијама карактера које одговарају горе поменутим техникама.

3.1.2 Keyboard augmenter

Keyboard аугментер је техника за аугментацију карактера која симулира грешке приликом куцања на тастатури.

Функционише на следећи начин:

- **Случајно додавање/изостављање карактера**: Keyboard аугментер може случајно додати нове карактере у текст или изоставити постојеће карактере, симулирајући грешке које се могу догодити приликом куцања на тастатури.

- **Замена карактера**: Може заменити одређене карактере у тексту другим карактерима који су близу оригиналних на тастатури, као што су слова која се налазе поред оригиналних на тастатури.

- **Промена распореда карактера**: Могућа је промена распореда карактера у тексту како би се симулирали случајно притиснути тастери на тастатури, што може резултирати потпуно различитим речима или фразама.

- **Додавање шума**: Слично као и код OCP аугментера,keyboard аугментер може додати шум у текст симулирајући грешке приликом куцања, попут случајног притискања више тастера истовремено.

- **Симулација различитих тастатура:** Могуће је симулирати различите тастатуре, као што су QWERTY, AZERTY или Dvorak, како би се генерисали текстови који одражавају различите распореде карактера на тастатури.

Ове варијације могу бити корисне за обогаћивање тренинг скупа података и побољшање перформанси модела обраде текста у стварним условима где се могу јавити грешке приликом уношења текста.

3.1.3 Random Char Augmenter

Random Char Augmenter је техника за аугментацију карактера која случајно извршава промене на карактерима текста.Постоје четири основне операције које Random Char Augmenter може извршити:

- **Insert** (*Уметање*): Ова операција случајно додаје додатне карактере на насумичним местима у тексту.

- **Substitute** (*Замена*): Ова операција случајно замењује постојеће карактере другим карактерима.

- **Swap** (*Замена места*): Ова операција случајно мења места између два карактера у тексту.

- **Delete** (*Брисање*): Ова операција случајно брише постојеће карактере из текста.

3.2 Аугментација речи

Аугментација речи је есенцијална техника у обради природног језика која доприноси обогаћивању скупа података кроз различите модификације речи у текстуалном корпусу.Кроз ову врсту аугментације, модели могу научити да препознају шире спектар варијација речи и контекста у којем се користе, што их чини способнијим за разумевање и генерисање природног језика.

У наставку ћемо прегледати неколико кључних техника аугментације речи.

3.2.1 Spelling augmenter

Spelling augmenter је техника која се користи за унапређење обраде природног језика кроз симулацију правописних грешака. Ова техника мења исправне речи у тексту са њиховим варијантама које су често резултат грешака приликом куцања.

Функционише на основу речника који садржи правописне грешке или алтернативне облике речи. Када се примени на текст, може значајно променити садржај, што може бити корисно у различитим ситуацијама. На пример, у процесу обуке модела за обраду природног језика, додавање синтетичких грешака може унапредити перформансе модела у препознавању и исправљању правописних грешака.

Међутим, важно је пажљиво применити ову технику, посебно у ситуацијама где је тачност од суштинског значаја. Неправилна примена Spelling augmentera може резултирати генерисањем нелогичних или погрешних речи, што може нарушити квалитет текста или анализе резултата.

3.2.2 Synonym augmenter

Synonym аугментер је техника која се користи како би се речи у тексту замениле њиховим синонимима. Ово се може постићи коришћењем различитих извора синонима, а један од најчешћих је **WordNet**, лексичка база података која садржи синониме, антониме и друге језичке информације.

Осим WordNeta, постоје и друге врсте извора синонима које се могу користити:

- **Paraphrase Database (PPDB):** Велика база података која садржи реченице или фразе које имају слично или исто значење. Основни циљ PPDBa је пружање парафраза, тј. различитих израза или формулација који преносе исто или слично значење.

- **Word Embeddings**: Векторско угнежђивање речи (word embeddings) може се користити за проналажење сличних речи у простору угнежђивања речи. Синоними се могу идентификовати претрагом вектора речи који су блиски вектору циљне речи.

- **Контекстуално учење**: Неки модели засновани на дубоком учењу могу научити значење речи и њихове везе кроз контекстуално учење. Ови модели могу генерисати синониме на основу образаца у корпусу текста на којем су обучени.

Разлике између ових извора синонима могу се односити на обим и квалитет синонима које пружају, као и на могућност прилагођавања специфичним потребама или доменима текста. На пример, WordNet је добро структуриран и пружа прецизне синониме, док word embeddings могу пружити више контекстуално релевантних синонима. Међутим, разноликост и прецизност синонима могу варирати у зависности од извора и квалитета података.

Поред Synonym augmentera, такође имамо и **Antonym augmenter**,ова техника проналази антониме, тј. речи које имају супротно значење у односу на дату реч, и замењује оригиналне речи у реченици са њиховим антонимима.

3.2.3 Random word augmenter

Random word augmenter је техника која укључује различите акције као што су замена, брисање и исецање речи у оригиналном тексту:

- **Случајна замена:** Ова акција подразумева замену одређених речи или токена у оригиналном тексту са насумично одабраним речима из истог језичког скупа. Ова замена додаје разноликост у текст, али може резултирати губитком смисла или стварањем нелогичних реченица.

- **Случајно брисање:** Ова акција укључује насумично уклањање одређеног броја речи из оригиналног текста. Прво се насумично бирају речи које ће бити уклоњене, а затим се оне избацују из текста. Ово може додати шум и разноликост у текст, али може довести до губитка информација и потребе за пажљивим прилагођавањем параметара како би се сачувао основни смисао текста.

- **Crop:** Ова акција се односи на процес уклањања одређеног броја речи из текста на насумично одабраним местима. Када се користи овај поступак, одређени део текста се једноставно "одсече" или уклони.

3.2.4 Split augmenter

Split augmenter је техника која се користи за раздвајање речи на два токена на насумично одабраним местима. Овде се појам "токен" односи на део текста који је раздвојен или подељен на два дела на насумично одабраним местима. Ово значи да се реч која је првобитно била један токен, сада дели на два токена.

Пример:

Оригинална реченица: "Дечак вози бицикл."

Split augmenter може да дели реч "вози" на два токена, на пример "во" и "зи", резултирајући реченицом: "Дечак во бицикл зи."

3.2.5 Contextual word embeddings augmenter

Contextual Word Embeddings Augmenter је техника која користи контекстуална угнежђивања речи (word embeddings) као основу за додавање или замену речи у тексту.
Word embeddings су репрезентације речи у векторском формату који се користе у обради природног језика. Свака реч се мапира у вектор у вишедимензионалном простору, при чему се сличне речи налазе близу једна другој.

Контекстуална угнежђивања речи додају додатни слој сложености тако што узимају у обзир контекст око речи приликом генерисања репрезентација. То значи да се значење речи може разликовати у зависности од контекста у којем се појављује.Ова репрезентација омогућава моделима да боље разумеју семантичке везе између речи.

Користи напредне моделе као што су BERT, DistilBERT, RoBERTa или XLNet за генерисање контекстуално обогаћених репрезентација речи у тексту.

BERT (Bidirectional Encoder Representations from Transformers) је један од најпознатијих модела који користи контекстуална угнежђивања речи. BERT је развио Google AI и заснован је на Трансформер архитектури. Он има способност да узима у обзир контекст око сваке речи у тексту приликом генерисања репрезентација речи, што му омогућава да разуме сложене семантичке везе и нијансе у језику.

DistilBert је компактнија верзија BERT модела која је развијена ради ефикасније употребе ресурса приликом обуке и коришћења модела. DistilBert задржава већи део перформанси BERT модела, али са мањим бројем параметара. То га чини погодним избором за апликације где је ресурсно ефикасно решење важно.

RoBERTa (Robustly Optimized BERT approach) је још један напредни модел који користи Трансформер архитектуру, али са побољшањима у односу на оригинални BERT. RoBERTa је оптимизован за обраду природног језика на различитим језицима и показује боље перформансе у неким задацима у поређењу са BERT моделом.

XLNet је модел заснован на Трансформер архитектури који користи идеју permuted language modeling-a. Ова техника омогућава моделу да узима у обзир све пермутације речи у реченици приликом генерисања репрезентација речи, што му омогућава да боље разуме сложене контекстуалне везе у језику.

Сваки од ових модела има своје предности и мане у зависности од специфичних захтева апликације. BERT се често користи због своје популарности и широке примењивости, док се DistilBert често користи када је потребна ефикасност у ресурсима. RoBERTa и XLNet такође имају своје специфичне примене у различитим контекстима. Одабир модела зависиће од конкретних захтева и карактеристика проблема који се решава.

Када се користе за аугментацију текста, контекстуална угнежђивања речи могу се користити на неколико начина:

- **Insert:** Уметање речи у текст користећи контекстуална угнежђивања речи. Ово укључује додавање нових речи у реченице на начин који је семантички прихватљив у датом контексту.

- **Substitute:** Замена речи у тексту користећи контекстуална угнежђивања речи. Ова акција подразумева замену постојећих речи у тексту са другим речима које имају слично значење у датом контексту.

Користећи напредне моделе као што су BERT, DistilBert, RoBERTa или XLNet за генерисање контекстуалних угнежђивања речи, Contextual Word Embeddings Augmenter омогућава обогаћивање текста на начин који узима у обзир шири контекст и семантичке везе међу речима. Ово може резултирати генерисањем нових варијација текста које су семантички богатије и боље прилагођене оригиналном контексту.

3.3 Flow

Flow у обради текста омогућава повезивање и примену вишеструких техника аугментације текста. Он се дели на два главна типа: sequential and sometimes pipelines.

Sequential flow је тип flow-a у обради текста где се трансформације примењују једна за другом у одређеном редоследу. Овај приступ омогућава дефинисање серије трансформација које се примењују на текст у одређеном распореду, при чему свака трансформација доприноси промени текста на одређени начин. На пример, у sequential flow-u, текст може проћи кроз серију трансформација као што су замена речи синонимима, додавање шума, избацивање речи итд., при чему се свака трансформација примењује након претходне.

Sometimes flow је варијанта flow-a у обради текста која се користи када није увек потребно примењивати све трансформације већ када се жели случајно изабрати између више трансформација. Овај приступ омогућава флексибилност у одабиру које трансформације ће бити примењене на текст у свакој итерацији, што може допринети већој разноврсности и непредвидљивости у генерисању аугментираних текстова. На пример,могуће је дефинисати сет трансформација као што су замена речи синонимима, додавање шума, избацивање речи итд., али сваки пут када се примењује flow, само неке од тих трансформација ће бити примењене, док ће остале бити прескочене.

3.4 Аугментација реченица

Аугментација реченица се фокусира на генерисање нових реченица из постојећих текстуалних података. Она се разликује од аугментације карактера и речи по томе што се фокусира на трансформацију целих реченица уместо појединачних карактера или речи.Ова техника има за циљ генерисање нових реченица које задржавају суштину оригиналних, али са промењеном структуром.

3.4.1 GPT2 i XLNet

GPT-2 и XLNet су два позната модела дубоког учења који су се показали ефикасним за различите задатке у обради природног језика. Оба модела се користе за генерисање нових текстуалних података, укључујући и аугментацију реченица.

GPT-2, развијен од стране OpenAI-a, користи се за генерисање текста на основу обимних скупова података на којима је трениран. Овај модел је способан да генерише нове реченице које су сличне онима које је видео током обуке. То значи да GPT-2 може генерисати нове реченице из постојећих реченица, задржавајући њихову суштину, али са промењеном структуром или изражавањем.

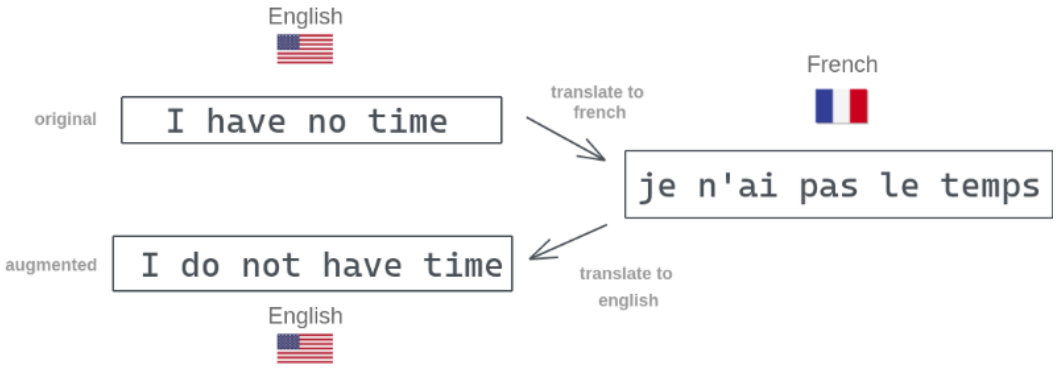
Са друге стране, XLNet, који је развила Google AI Language,користи permuted language modeling стратегију, што омогућава моделу да учи зависности између свих речи у тексту, не само претходних речи у низу. Ово омогућава XLNet-у да боље разуме контекст и да генерише кохерентније текстуалне податке.

Предности коришћења GPT-2 и XLNet за аугментацију реченица укључују њихову способност генерисања природног текста и прилагодљивост различитим језичким стиловима. Међутим, важно је имати на уму да ови модели могу имати тенденцију да генеришу неправилне или нелогичне реченице, као и могућност да произведу садржај који није у складу са оригиналним намерама.

3.5 Back translation

У овој методи, текстуалне податке преводимо на одређени језик, а затим их поново преводимо на оригинални језик. Ово може помоћи у генерисању текстуалних података са различитим речима, док се истовремено задржава контекст текстуалних података.

Ова техника је посебно корисна за генерисање тренинг података за класификационе задатке који садрже локализоване или двојезичне фразе, где је циљни језик нискоресурсни језик са мање доступних података.



Изазови и потенцијалне мане ове технике:

- **Губитак оригиналног значења**: Приликом превођења текста на други језик и поновног превода натраг на оригинални језик, може доћи до губитка суптилних нијанси и контекста оригиналног текста. Ово може резултирати генерисањем текста који није потпуно исти као оригинални текст.

- **Квалитет превода**: Квалитет превода зависи од квалитета коришћених преводилачких алата и модела. Неке речи или фразе могу бити погрешно преведене, што може довести до нетачног генерисања текста.

- **Ограничења преводилачких алата**: Преводилачки алати могу имати ограничења у подржаним језицима или квалитету превода за одређене језичке парове. На пример, неки мање заступљени језици могу имати мање доступне ресурсе за превод, што може ограничити ефикасност технике.

- **Трошкови и ограничења АПИ-ја**: Коришћење преводилачких АПИ-ја може бити повезано са трошковима, посебно ако се користи велика количина текста.Осим тога, неки преводилачки АПИ-ји могу имати ограничења у броју захтева или количини текста који се може превести, што може ограничити скалабилност ове технике.

- **Временски захтеви**: Back translation процес може бити временски захтеван, посебно ако се ради са великим количинама текста. Потребно је време за превођење текста на други језик, као и за поновно превођење назад на оригинални језик.

4 Практични део рада

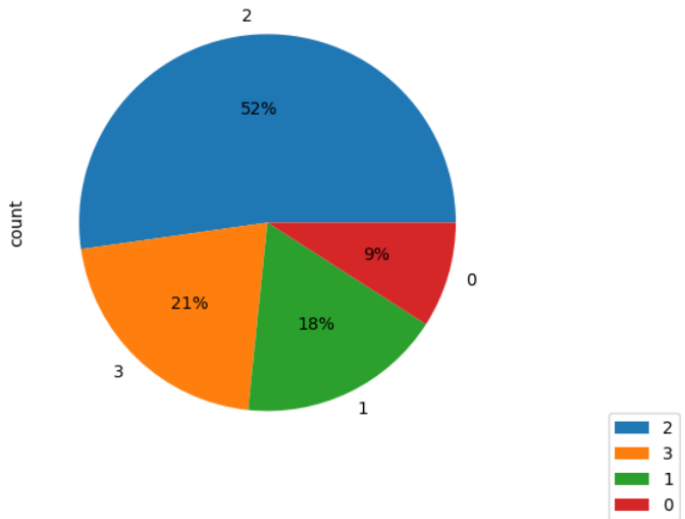
У практичном делу рада истражујемо утицај различитих техника аугментације текста на перформансе класификационих модела и дубоких неуронских мрежа користећи скуп података са твитовима о климатским променама: Twitter-climate-change-sentiment-dataset.

Скуп података садржи твитове који се односе на тему климатских промена, сакупљене у периоду од 27. априла 2015. до 21. фебруара 2018. године. Укупно је анотирано 43.943 твитова, при чему је сваки твит прегледан од стране три рецензента. Само твитови на које су се сви рецензенти сложили су задржани, док су остали одбачени, чиме је обезбеђен висок квалитет анотација.

Свако мишљење у твитовима је сврстано у једну од следећих категорија:

- **2 (Вести):** твитови који садрже линкове ка чињеницама о климатским променама
- **1 (Про):** твитови који подржавају веровање да су климатске промене резултат људских активности
- **0 (Неутрално):** твитови који не подржавају нити оспоравају веровање у антропогене климатске промене
- **-1 (Анти):** твитови који не верују у антропогене климатске промене

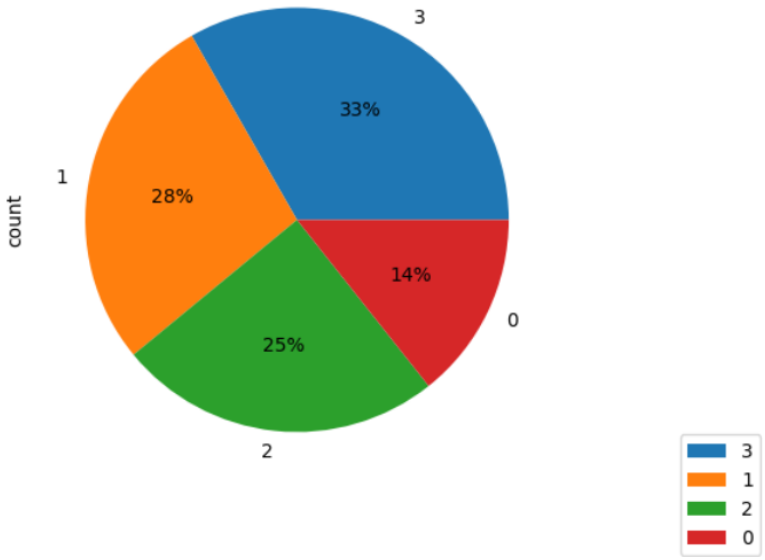
На почетку након учитавања скупа података и основне предобраде као што је преименовање колона,проверу типа података и недостајућих вредности погледали смо и балансираност скупа:



Слика 3. Балансираност почетног скупа података

Вршимо балансирање скупа података тако што смањујемо број примерака класе која има ознаку 2 (тј. оне који припадају категорији "Вести") за 70%. Ово радимо из два разлога: да бисмо уравнотежили скуп података и смањили његову величину због ограничења у обради података.

Сада скуп података има 27870 твитова и балансираност изгледа овако:



Слика 4. Балансираност крајњег скупа података

Затим уклањамо 20% редова који припадају класи са ознаком 3 из скупа података и коначан скуп података има укупно 26015 твитова.

Пре него што кренемо на примену техника за аугментацију текста упознаћемо се са бибилиотеком која је коришћена у практичном делу рада.

nlpaug је библиотека за аугментацију текста која пружа различите технике за генерисање нових текстуалних инстанци из постојећих података. Ова библиотека омогућава корисницима да прошире своје скупове података текстом помоћу различитих приступа, укључујући карактер, реч, реченицу и ток текста.

- **nlpaug.augmenter.char**: Ова компонента библиотеке омогућава аугментацију текста на нивоу карактера. То значи да се постојећи текстуални подаци мењају додавањем, изменом или брисањем карактера како би се генерисали нови текстови.

- **nlpaug.augmenter.word**: Ова компонента омогућава аугментацију текста на нивоу речи. То значи да се постојећи текстуални подаци мењају променом или заменским речима, додавањем синонима или реорганизацијом речи унутар реченица.

- **nlpaug.flow**: Ова компонента омогућава креирање сложених токова аугментације текста који комбинују више техника аугментације како би се генерисали разноврснији текстови.

- **nlpaug.augmenter.sentence**: Ова компонента омогућава аугментацију текста на нивоу реченица. То значи да се постојеће реченице могу изменити додавањем, изменом или брисањем речи, променом синтаксе или семантичким променама како би се генерисали нови текстови.

Сада анализу делимо на два дела у једном ће бити испитан утицај аугментације на перфомансе класификационих модела док ће у другом делу бити испитан утицај аугментације на перфомансе дубоких неуронских мрежа

4.1 Утицај аугментације на перфомансе класификационих модела

На почетку делимо скуп података на train (80%) и test податке (20%),затим као што смо објаснили у другом поглављу овог рада,потребно је да припремимо текст за аугментацију. Овај процес укључује неколико корака: чишћење текста, уклањање непотребних елемената, токенизацију и лематизацију.

Први корак је чишћење текста, што укључује уклањање HTML ознака, корисничких имена и URL-ова, како би текст био чистији и лакши за обраду. Емотикони, који носе емоционалну информацију, идентификују се и привремено складиште како би се вратили у текст након уклањања не-речних карактера и претварања свих речи у мала слова.

Следећи корак је токенизација текста, која подразумева раздвајање текста на појединачне речи или токене, што омогућава даљу обраду као што су уклањање стоп речи и лематизација.

Стоп речи, које немају значајан утицај на значење текста, уклањају се како би се смањио шум и побољшала ефикасност модела.

На крају, лематизација своди речи на њихов основни или коренски облик, чиме се смањује разноврсност облика речи и олакшава моделу разумевање текста.

Након претпроцесирања, текст се претвара у нумеричке векторе коришћењем Bag-of-Words модела, који узима у обзир учесталост појављивања речи и ствара речник са најфреквентнијим речима.

Овим процесом припреме текста обезбеђујемо да су текстуални подаци у стандардизованом и структурираном формату, што омогућава бољу примену техника аугментације и постиже се већа прецизност и поузданост модела у каснијим фазама анализе и класификације.

Затим, користимо TF-IDF трансформацију како бисмо припремили податке за моделирање.

TF-IDF (Term Frequency - Inverse Document Frequency) је широко коришћен статистички метод у обради природног језика и преузимању информација. Он мери колико је битан одређени термин унутар документа у односу на целокупан корпус докумената.

- **Term Frequency (TF):** TF представља број појављивања термина унутар документа у односу на укупан број речи у том документу. На пример, ако се реч појави пет пута у документу од 100 речи, TF те речи је 5/100 = 0.05.

- **Inverse Document Frequency (IDF**): IDF мери колико је термин специфичан за мали број докумената у корпусу. Ако се термин појављује у мањем броју докумената, добија већу вредност IDF, што значи да је термин ређи и самим тим важнији. Формула за IDF је логаритамски заснована и изгледа овако: IDF(term) = log(H/df), где је H укупан број докумената у корпусу, а df број докумената који садрже тај термин.

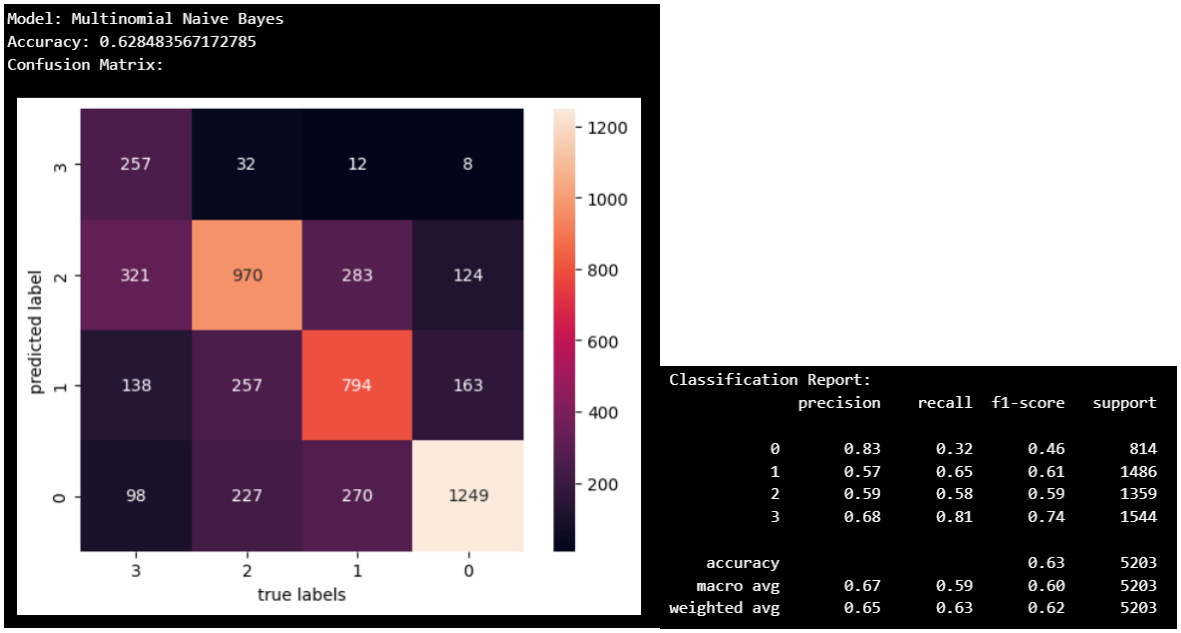
- **TF-IDF:** Композитни скор се добија множењем TF и IDF вредности, што даје меру важности термина која узима у обзир и његову учесталост у документу и реткост у корпусу.

У коду TfidfTransformer се иницијализује са параметрима use_idf=True, што значи да ће се користити IDF компонента, norm='l2' за L2 нормализацију (што помаже да се скалирају вектори тако да имају једнаку дужину), и smooth_idf=True за глатко IDF вредности (што спречава дељење са нулом за термине који се не појављују у тренинг корпусу).

Затим, применом fit_transform на тренинг податке и transform на тест податке, добијамо TF-IDF представу наших текстуалних података. Ова трансформација омогућава моделима да боље разумеју и важност термина унутар докумената и њихову реткост у целокупном корпусу, што води до бољих перформанси у задацима класификације.

Сада инцијализујемо речник models, у коме наводимо класификационе моделе,у овом случају имамо само један - "Multinomial Naive Bayes". Овај модел је одабран за анализу зато што је често коришћен у задацима класификације текста и обично даје добре резултате на текстуалним подацима. Могуће је касније проширење и додавање више модела уколико имамо довољно хардверских ресурса.

Након тога вршимо анализу без аугментације и ово су резлтати који су добијени:



Слика 5. Резултати без примене аугментације података

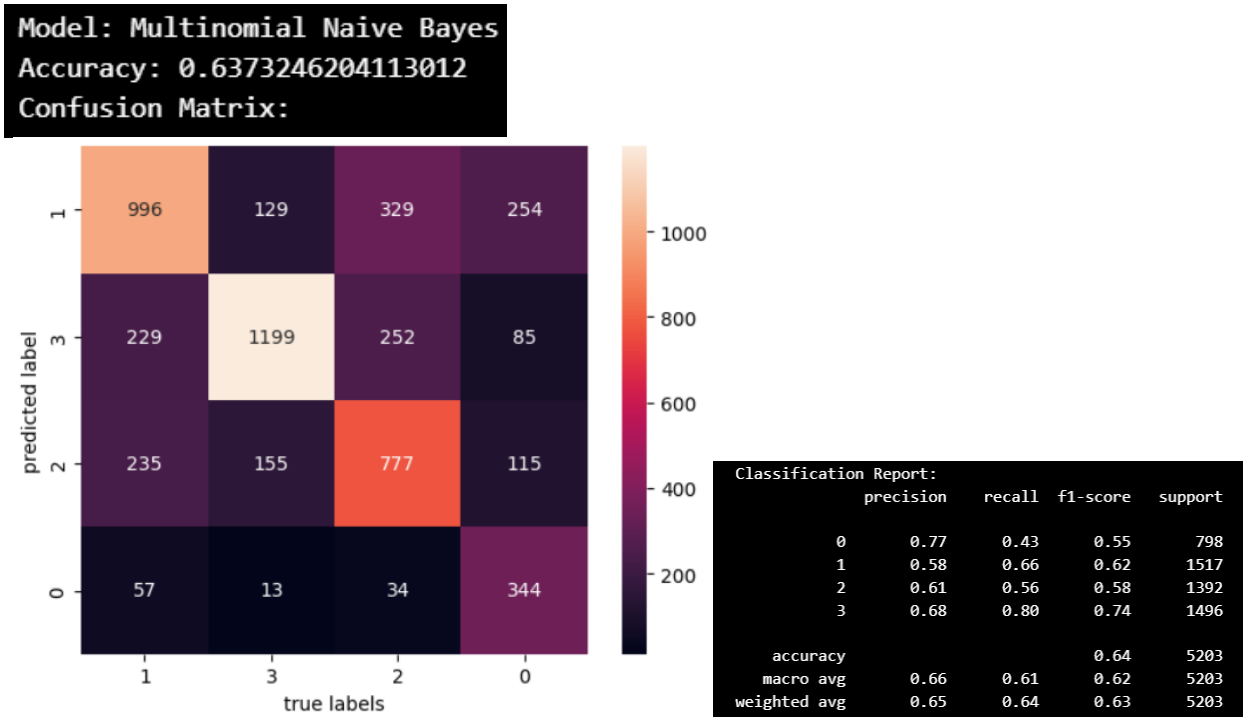
Да бисмо извршили евалуацију техника за аугментацију, користили смо функцију evaluate_aug, која прима следеће параметре: aug_strategy (стратегија аугментације), n (број аугментираних твитова по оригиналном твиту), X_train и y_train, као и X_test и y_test. У овој функцији, прво се генеришу аугментирани твитови на основу стратегије аугментације, затим се они додају оригиналним твитовима у тренинг скупу података. Након тога се примењују препроцесирање, токенизација и лематизација на текстуалне податке, а затим се граде и трансформишу у векторску репрезентацију текста. На крају, фитујемо модел на аугментираним тренинг подацима и евалуирамо га на тест подацима.

У овом истраживању смо експериментално испробали три технике за *аугментацију текста на нивоу карактера:*

- OCR Augmenter,
- Keyboard Augmenter
- Random Augmenter
    - Insert
    - Substitute
    - Swap
    - delete

Детаљно смо описали сваку од ових техника у поглављу 3 овог рада.

Након спровођења експеримената, утврђено је да је OCR Augmenter постигао најбоље резултате у смислу побољшања перформанси класификационог модела.



Слика 6. Резултати након примене аугментације карактера

Конкретно,техника је резултирала тачношћу класификације од 0.63732, док је без коришћења аугментације тачност износила 0.628. Ово указује на то да је примена OCR Augmenter резултирала побољшањем тачности класификационог модела за око 0.009.

Затим што се тиче *техника за аугментацију текста на нивоу речи* испробане су следеће технике:

- Spelling Augmenter
- Synonym Augmenter
    - Substitute word by WordNet's synonym
- Antonym Augmenter
- Random word augmenter
    - Swap
    - Delete
- Split augmenter
- Contextual Word Embeddings Augmenter
    - Insert (Bert)
    - Substitute(Roberta)

Најбоље резултате је дао Contextual Word Augmenter који користи BERT.Коришћење контекстуалних репрезентација речи из BERT-а омогућава генерисање нових реченица које су семантички сличне оригиналном тексту, чиме се очувава суштина података. BERT је обучен на великом корпусу текста и способан је да научи сложене језичке обрасце, што му омогућава да боље разуме значење речи у контексту реченице.

Осим тога, BERT може генерисати реалистичне податке јер балансира између разноликости и квалитета. То значи да новонастали текстови нису само разноврсни, већ и семантички тачни, што је кључно за успешну аугментацију текста. Такође, BERT укључује контекстуално учење речи, што му омогућава да ухвати нијансе значења речи у различитим контекстима.

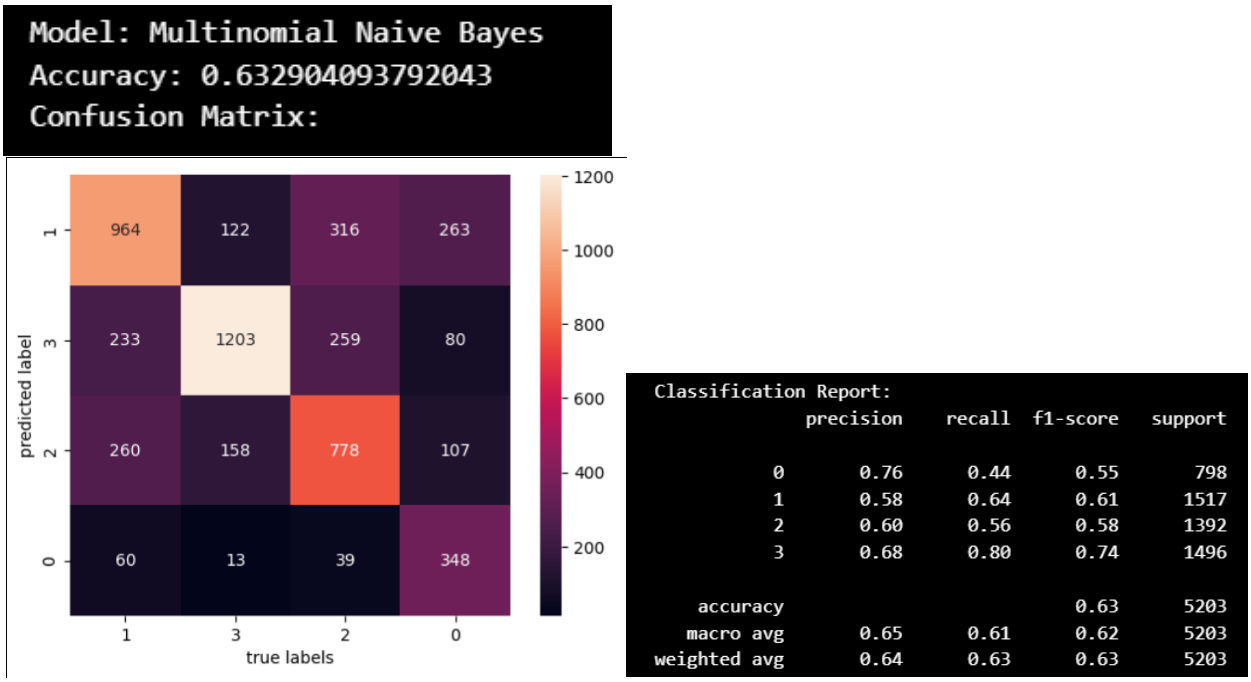
Затим сам користила *flow* за генерисање речи користећи библиотеку nlpaug.flow. У секвенцијалној (Sequential) pipeline, користила сам RandomCharAug за убацивање и RandomWordAug за замену речи. С друге стране, у pipeline Sometimes сам комбиновала RandomCharAug за брисање и RandomCharAug за убацивање, заједно са RandomWordAug. Међутим ове технике нису значајно побољшале перформансе у односу на почетни корпус без аугментације,али остављају доста могућности за експериментисање.

На крају испробана је и *аугментација реченица* користећи Contextual Word Embeddings базиран на моделу GPT-2. Овај модел, који припада породици трансформатора, дизајниран је за генерисање текста и има моћну способност разумевања контекста унутар реченица.

GPT-2 (Generative Pre-trained Transformer 2) је модел дубоког учења који је трениран на огромном корпусу текста са интернета. Његова архитектура омогућава моделирање дугорочних зависности у тексту, што га чини изузетно способним за задатке генерисања природног језика. GPT-2 може да предвиди следећу реч у реченици на основу претходних речи, што му омогућава да генерише кохерентан и смислен текст.

Contextual Word Embeddings користи ове способности модела GPT-2 да генерише нове реченице које су семантички сличне оригиналним реченицама. Ова техника омогућава аугментацију реченица тако што ствара варијанте постојећих реченица, задржавајући њихово основно значење али додајући разноликост у тренинг податке.

Резултати који су добијени овом методом су следећи:



Слика 7. Резултати након примене аугментације реченица

Важно је напоменути да смо у овом истраживању користили само n = 1, односно генерисали смо само један аугментирани твит за сваки оригинални твит. Уколико бисмо користили већи број аугментираних твитова (веће n), разлика у перформансама модела могла би бити још израженија.

4.2 Утицај аугментације на перфомансе дубоких неуронских мрежа

На почетку овог процеса, подаци се деле на три скупа: тренинг (80%), валидациони (10%), и тест (10%) скуп користећи train_test_split из sklearn библиотеке. Ово осигурава да имамо одвојене скупове података за тренирање, валидацију и тестирање модела, што је кључно за процену његове перформансе.

Затим, сваки од ових DataFrame-ова се конвертује у Dataset у Aparche Arrow формату користећи Dataset.from_pandas. Ови Datasets се затим окупљају у један DatasetDict, који омогућава лакшу манипулацију и руковање подацима.

Након овог корака, добија се структура података која изгледа овако:

```
DatasetDict({
    train: Dataset({
        features: ['label', 'text'],
        num_rows: 20812
    })
    val: Dataset({
        features: ['label', 'text'],
        num_rows: 2601
    })
    test: Dataset({
        features: ['label', 'text'],
        num_rows: 2602
    })
})
```

Слика 8. Структура података

Ова структура омогућава да се подаци лако поделе и користе за тренирање, валидацију и тестирање модела дубоких неуронских мрежа.

Даље, одређује се контролна тачка модела (model checkpoint), у овом случају "distilbert-base-uncased", која ће се користити за учитавање пре-тренираног модела. Затим се учитава токенизатор из пре-тренираног модела помоћу AutoTokenizer.from_pretrained.

У овом контексту, коришћење distilbert-base-uncased као модел контролне тачке (model checkpoint) значи да ћемо користити пре-тренирани модел DistilBERT за класификацију текстова. Ево детаљнијег објашњења зашто се користи DistilBERT и шта значи модел контролна тачка:

Зашто DistilBERT?

- Ефикасност: DistilBERT је компримована верзија BERT-а, која је око 40% мања и бржа, а задржава око 97% перформанси у поређењу са оригиналним BERT моделом. Ово га чини идеалним за употребу у апликацијама где је брзина критична или где постоје ограничења у меморији.

- Прецизност: Иако је мањи, DistilBERT и даље пружа високу прецизност и перформансе за разне задатке обраде природног језика, укључујући класификацију текста, чинећи га погодним за нашу примену.

- Претходно трениран: Користећи пре-тренирани модел као што је DistilBERT, можемо искористити знање које је модел стекао током тренинга на великом корпусу текста. Ово значајно смањује време и ресурсе потребне за тренирање модела од нуле и омогућава бржу примену за специфичне задатке.

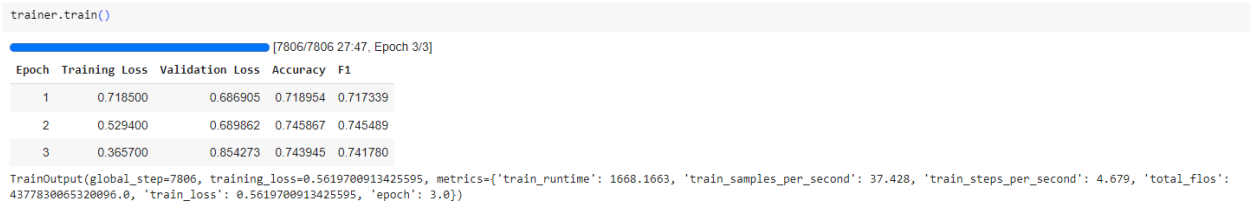
Затим се приступа токенизацији целог скупа података. За ово користимо map() метод нашег DatasetDict објекта, што омогућава примену функције за обраду на сваки елемент у скупу података. Пошто модел очекује тенсоре као улазне податке, конвертујемо колоне input_ids и attention_mask у "torch" формат.

Следећи корак је учитавање претходно обученог DistilBERT модела. Користимо AutoModelForSequenceClassification уместо AutoModel из Hugging Face Transformers библиотеке. Разлика је у томе што AutoModelForSequenceClassification има класификациону главу на врху претходно обучених излаза модела, што омогућава лако тренирање са базним моделом. Потребно је само специфицирати колико ознака (класа) модел треба да предвиди (у нашем случају четири), што одређује број излаза класификационе главе.

За праћење метрика током тренинга, дефинише се compute_metrics() функција за Trainer класу. За наш случај, израчунавају се F1-скор и тачност модела.

Да бисмо дефинисали параметре тренинга, користимо TrainingArguments класу. Најважнији аргумент је output_dir, који одређује где ће се чувати сви параметри тренинга. Параметри тренинга укључују величину серије, број епоха, стопу учења, тежинску деконцентрацију за регуларизацију, стратегију евалуације и друге. Тренинг аргументи такође укључују подешавање да се најбољи модел учита након завршетка тренинга и да се модел не шаље на Huggingface Hub због временских ограничења.

Затим дефинишемо тренера користећи Trainer класу која покреће процес тренинга. Ова класа укључује модел, параметре тренинга, метрике, скупове података за тренинг и валидацију, и токенизатор. Покреће се процес тренинга и прате се метрике.

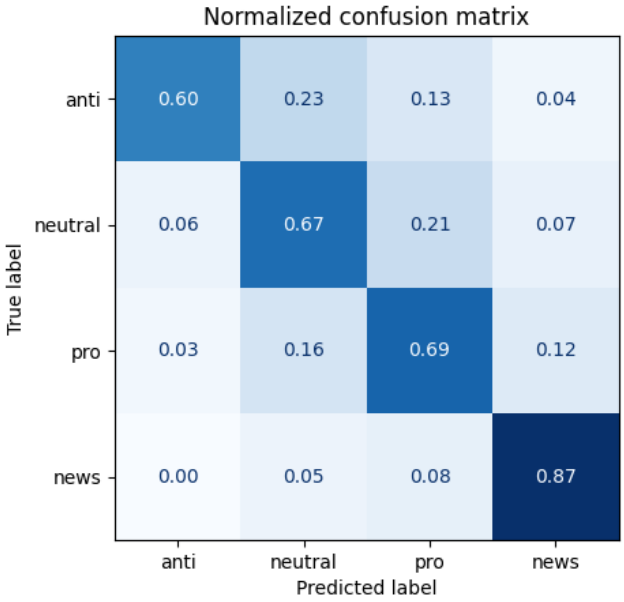


Слика 9. Резултати тренинга без примене аугментације података

Резултати тренинга се могу детаљно анализирати израчунавањем матрице конфузије. Да би се визуализовала матрица конфузије, прво се добијају предвиђања на валидационом скупу користећи predict() метод Trainer класе. Овај метод враћа објекат PredictionOutput који садржи низове предвиђања и label_ids, заједно са метрикама које смо дефинисали.

```
{'test_loss': 0.6869049668312073,
 'test_accuracy': 0.7189542483660131,
 'test_f1': 0.7173390861965514,
 'test_runtime': 7.029,
 'test_samples_per_second': 370.039,
 'test_steps_per_second': 46.379}
```

Матрица конфузије за валидациони скуп:



Поред тога, анализирају се и метрике перформанси модела на тест скупу података како би се добила свеобухватна слика о способности модела да генерализује на новим подацима.

```
{'test_loss': 0.6743456721305847,
 'test_accuracy': 0.7355880092236741,
 'test_f1': 0.7325217424086723,
 'test_runtime': 7.6792,
 'test_samples_per_second': 338.838,
 'test_steps_per_second': 42.452}
```

На крају процеса, након што је модел обучен, потребно је сачувати најбољи модел локално. Да би се модел сачувао користимо save_pretrained метод и чувамо га у одређеном директоријуму (distilbert-base-uncased-finetuned-tweets-climate-change).

Затим учитавамо локално сачувани најбољи модел и токенизатор. За ово користимо pipeline из transformers библиотеке, која нам омогућава да лако креирамо класификатор за текст користећи наш претходно обучени модел и токенизатор.

Након тога, можемо тестирати наш класификатор са примером твита. На пример, користимо твит "Stop dumping trash and save our ocean!" и добијамо предвиђања класификатора:

```
[[{'label': 'anti', 'score': 0.03508559241890907},
  {'label': 'neutral', 'score': 0.20993320643901825},
  {'label': 'pro', 'score': 0.7516967058181763},
  {'label': 'news', 'score': 0.0032845111563801765}]]
```

Такође, можемо тестирати класификатор са другим твитом, као што је "Global warming is a fake claim made by some crazy scientists!" и видети како класификатор реагује на различите врсте твитова. Ово нам омогућава да проверимо како модел ради на новим, невежбаним подацима и добијемо повратне информације о његовим перформансама.

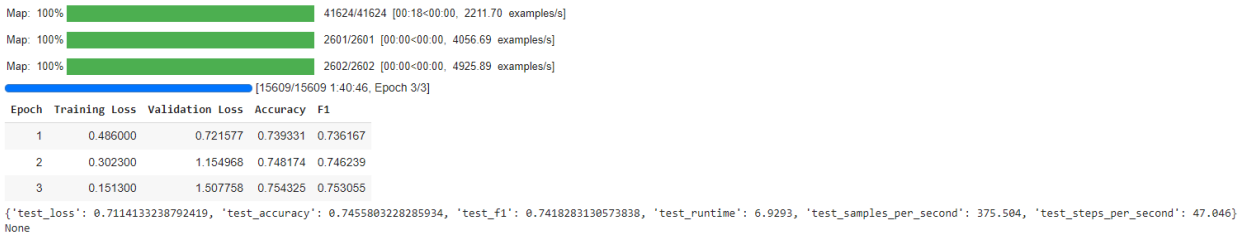
```
[{'label': 'anti', 'score': 0.9285029172897339}]
```

Затим прелазимо на аугментацију текста и њихов утицај на дубоке неуронске мреже.Користимо функцију `evaluate_aug1` да би видели еваулацију перфомански модела. Прво, генеришу се аугментирани твитови за сваки твит у тренинг сету, а затим се аугментирани твитови и њихове ознаке чувају у листама. Оригинални и аугментирани подаци се затим комбинују у нови тренинг сет.Након тога, нови тренинг сет се токенизује и конвертује у тензоре које модел може да обради. Модел се тренира на овом проширеном тренинг сету, а меморија се чисти како би се избегли проблеми са недостатком меморије током тренинга. На крају, модел се евалуира на тест сету, а резултати се исписују.

Биће обрађена три типа аугментације: аугментација карактера, речи и реченица.

Због комплексних операција које дуго трају, као што је тренинг модела дубоких неуронских мрежа, за сваку од техника аугментације испробана је само по једна метода. Методу сам изабрала на основу тога која је давала најбоље резултате у анализи перформанси класификационих модела. Тренинг дубоких неуронских мрежа захтева значајне хардверске ресурсе, због чега је овај део рада изведен у Google Colab окружењу, које пружа приступ T4 GPU-у. Међутим, иако Google Colab нуди моћне ресурсе, његово коришћење је ограничено временом трајања сесије (runtime). Ова ограничења су утицала на обим и дужину тренинга модела, али су ипак омогућила довољно ресурса за евалуацију ефеката различитих метода аугментације текста на перформансе дубоких неуронских мрежа.

Због тога за *аугментацију карактера* изабран је **OCR augmenter**,и резултати који су добијени овом методом су следећи:



Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.486000	0.721577	0.739331	0.736167
2	0.302300	1.154968	0.748174	0.746239
3	0.151300	1.507758	0.754325	0.753055

Слика 10. Резултати тренинга са применом аугментације карактера

Анализом резултата модела без аугментације и модела са аугментацијом путем OCR augmentera могу се уочити значајне разлике у перформансама модела.

Training Loss је био значајно нижи код модела са аугментацијом. Почетни Training Loss је био 0.486000 и смањио се на 0.151300 до краја тренинга, док је код модела без аугментације почетни Training Loss био 0.718500 и смањио се на 0.365700. Овај пад Training Loss-а указује да је модел са аугментацијом брже и ефикасније учио током тренинга.

Међутим, **Validation Loss** је био већи код модела са аугментацијом. Почетни Validation Loss је био 0.721577 и повећао се на 1.507758 у трећој епохи, док је код модела без аугментације Validation Loss био најнижи у првој епохи (0.686905) и повећао се до 0.854273 у трећој епохи. Већи Validation Loss код модела са аугментацијом може указивати на оверфитовање, што значи да модел добро учи на тренинг подацима, али се теже прилагођава на валидационе податке.

У погледу **Accuracy**, модел са аугментацијом је имао највећу тачност у трећој епохи са 0.754325, док је модел без аугментације имао највећу тачност у другој епохи са 0.745867. Ово показује да аугментација може довести до побољшања тачности модела.

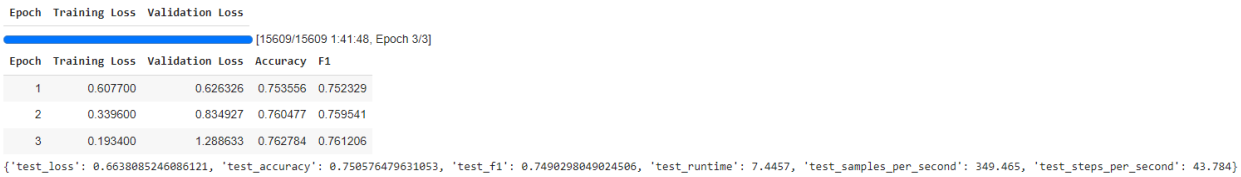
F1 скор је такође био бољи код модела са аугментацијом, са највишим резултатом у трећој епохи од 0.753055, у поређењу са моделом без аугментације који је имао највиши F1 скор од 0.745489 у другој епохи. Ово указује да је модел са аугментацијом боље балансиран у погледу прецизности и одзива.

Најважније је да **Тест Перформансе** показују да је модел са аугментацијом постигао боље резултате на невиђеним подацима. Test Loss је био 0.711413, Test Accuracy 0.745580, и Test F1 0.741828, што су бољи резултати у поређењу са Validation резултатима модела без аугментације у трећој епохи. Ово сугерише да аугментација података побољшава генерализацију модела, што значи да је модел боље прилагођен за рад са новим, невиђеним подацима.

Затим следећа техника је *аугментација речи*,овде су испробане две методе:

- Synonym Augmenter
  - Substitute word by WordNet's synonym
- Contextual Word Embeddings Augmenter
  - Insert (Bert)

Резултати:

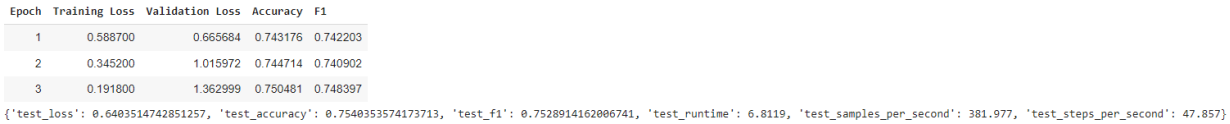


Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.607700	0.626326	0.753556	0.752329
2	0.339600	0.834927	0.760477	0.759541
3	0.193400	1.288633	0.762784	0.761206

{'test_loss': 0.6638085246086121, 'test_accuracy': 0.750576479631053, 'test_f1': 0.7490298049024506, 'test_runtime': 7.4457, 'test_samples_per_second': 349.465, 'test_steps_per_second': 43.784}

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.588700	0.665684	0.743176	0.742203
2	0.345200	1.015972	0.744714	0.740902
3	0.191800	1.362999	0.750481	0.748397

Слика 10. Резултати тренинга са применом аугментације речи (*Substitute word by WordNet's synonym)*



Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.588700	0.665684	0.743176	0.742203
2	0.345200	1.015972	0.744714	0.740902
3	0.191800	1.362999	0.750481	0.748397

{'test_loss': 0.6403514742851257, 'test_accuracy': 0.7540353574173713, 'test_f1': 0.7528914162006741, 'test_runtime': 6.8119, 'test_samples_per_second': 381.977, 'test_steps_per_second': 47.857}

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.588700	0.665684	0.743176	0.742203
2	0.345200	1.015972	0.744714	0.740902
3	0.191800	1.362999	0.750481	0.748397

Слика 11. Резултати тренинга са применом аугментације речи (*Contextual Word Embeddings Augmenter)*

Пошто свакако обе методе дају боље резултате у односу на класификацију без аугментације,упоредићемо их међусобно.

Губитак приликом обуке и валидације:

- Оба приступа показују сличне трендове у губицима приликом обуке, али Contextual Word Embeddings Augmenter приступ показује ниже губитке током целог процеса обуке.

- Међутим, Substitute word by WordNet's synonym приступ показује мањи губитак приликом валидације у првој епохи, али нагли раст губитка у каснијим епохама, док Contextual Word Embeddings Augmenter приступ показује постепено повећање губитка током целог процеса.

Тачност и F1 скор:

- Contextual Word Embeddings Augmenter приступом постиже нешто већу тачност и Ф1 скор у трећој епохи, док Substitute word by WordNet's synonym приступ достиже максималне вредности у другој епохи, али се значајно погоршава у трећој епохи.

Тестирање:

- Contextual Word Embeddings Augmenter приступом показује боље резултате на тестирању, са нижим губитком, већом тачношћу и већим F1 скором у поређењу са Substitute word by WordNet's synonym приступом.

На основу ових анализа, можемо закључити да **Contextual Word Embeddings Augmenter** приступ има предност у погледу стабилности током обуке, као и бољих перформанси на тестирању, показујући потенцијал за бољу генерализацију модела у реалним условима.

Последња техника је *аугментација речи*,и овде је испробана метода Word Embeddings базиран на моделу GPT-2.

Резултати:

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.494900	0.669121	0.761630	0.759266
2	0.222000	1.144494	0.763168	0.760871
3	0.110300	1.507957	0.759323	0.757515

{'test_loss': 0.6830383539199829, 'test_accuracy': 0.754803996925442, 'test_f1': 0.752698878795083, 'test_runtime': 6.6182, 'test_samples_per_second': 393.159, 'test_steps_per_second': 49.258}

Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.494900	0.669121	0.761630	0.759266
2	0.222000	1.144494	0.763168	0.760871
3	0.110300	1.507957	0.759323	0.757515

Слика 12. Резултати тренинга са применом аугментације реченица

Анализа резултата показује да аугментација реченица (Word Embeddings базиран на моделу GPT-2) даје најбоље перформансе, постижући најнижи губитак током обуке и највишу тачност и F1 скор на тестирању. Ова техника се показала као најефикаснија у побољшању способности модела да генерализује и правилно класификује нове податке.

Међутим, важно је напоменути да су и аугментација карактера и аугментација речи показале значајна побољшања у односу на модел без примене аугментације. Ове технике, иако нешто мање ефикасне од аугментације са реченицама, ипак су допринеле повећању тачности и F1 скора модела, што указује на важност примене аугментације у процесу обуке модела.

5 Закључак

У истраживачком делу рада детаљно смо проучавали различите технике за аугментацију текста и њихов утицај на перформансе класификационих модела и дубоких неуронских мрежа. Почевши од процеса припреме текста за аугментацију, анализирали смо методологије као што су чишћење текста, нормализација, токенизација, лематизација и уклањање стоп-речи. Након тога, систематски смо прегледали различите технике аугментације текста, укључујући оне које се фокусирају на карактере, речи и реченице.

Кроз експериментални део рада, провели смо детаљну анализу утицаја ових техника на перформансе класификационих модела и дубоких неуронских мрежа. Резултати наших експеримената јасно су показали да аугментација текста може значајно побољшати перформансе ових модела.

Овај рад пружа чврсту основу за даља истраживања у области аугментације текста. Са напретком технологије и развојем нових метода дубоког учења, очекује се да ће аугментација текста постати још ефикаснија и широко примењивана техника у обради природног језика. Ово ће допринети унапређењу перформанси различитих NLP модела у будућности. Важно је истаћи да се ова област тренутно налази у почетној фази истраживања, али обрада природног језика бележи велики напредак и експанзију.

6 Литература

[1] https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

[2] KDnuggets. (2019, April). Text preprocessing in NLP & ML (with Python code)

[3] Packt Publishing. (2023). *Data Augmentation with Python: Enhance accuracy in Deep Learning with practical Data Augmentation for image, text, audio* (1st ed.).

[4] https://neptune.ai/blog/document-classification-small-datasets

[5] https://neptune.ai/blog/data-augmentation-nlp

[6] Badimala, P., Mishra, C., Modam Venkataramana, R. K., & Dengel, A. (2019). A Study of Various Text Augmentation Techniques for Relation Classification in Free Text.

[7] https://www.geeksforgeeks.org/text-augmentation-techniques-in-nlp/

[8] https://towardsdatascience.com/data-augmentation-library-for-text-9661736b13ff

[9] https://huggingface.co/

[10] Shorten, C., Khoshgoftaar, T., & Furht, B. (2021). Text Data Augmentation for Deep Learning. *Journal of Big Data*, *8*(1),

[11] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

[12] Khosla, C., & Saini, B. S. (Godina izdanja). Enhancing Performance of Deep Learning Models with different Data Augmentation Techniques.

[13] https://sunscrapers.com/blog/deep-learning-for-nlp-an-overview/