SZAKDOLGOZAT

SIMON-JÁRÓKA HELÉNA LILLA

Üzleti Adatelemző

Budapest

2024

Budapesti Corvinus Egyetem

Természetes Nyelvi Feldolgozás és Transzfer Tanulás

Belső konzulens: Dr. Racskó Péter

Szakfelelős: Dr. Fodor Szabina Eszter

Készítette: Simon Járóka Heléna Lilla

VPMNQN

Üzleti Adatelemző Szak

Szakirányú továbbképzés – esti

Budapest

2024

Tartalomjegyzék

1. BEVEZETÉS

1.1 A téma jelentősége és aktualitása

A mesterséges intelligencia (MI) gyors fejlődése az elmúlt évtizedben nagy hatással volt a természetes nyelvi feldolgozás (NLP) területére. Az NLP célja, hogy a számítógépek képesek legyenek megérteni, feldolgozni és generálni az emberi nyelvet. Az NLP gyakorlati alkalmazásai közé tartozik a gépi fordítás, chatbotok, szöveges tartalom elemzése és beszédfelismerés. Például a Google Translate naponta több mint 100 milliárd szót fordít le különböző nyelvek között, míg az Apple Siri és az Amazon Alexa NLP segítségével értik meg és válaszolják meg a felhasználói kéréseket (Durai, 2021)

1.2 A természetes nyelvi feldolgozás (NLP) meghatározása

A természetes nyelvi feldolgozás (NLP) a számítógépes tudomány azon ága, amely a számítógépek és az emberi nyelv közötti interakcióval foglalkozik. Az NLP alkalmazásai közé tartozik a szövegklasszifikáció, szentimentelemzés, beszédszintézis és beszédfelismerés. Ezek a technológiák lehetővé teszik az automatizált ügyfélszolgálati rendszerek és a személyre szabott tartalomajánlók működését (Shaikh R. , 2018)

„A természetes nyelven írt szövegek számítógépes feldolgozása (Natural Language Processing – NLP) a gépi tanulás és a nyelvtudomány közös területe. A természetes nyelv gépi eszközökkel történő „megértése” bonyolult feladat, nem elég a szavak és mondatok jelentésének, hanem a szövegkörnyezet és a kommunikáló felek ismerete is fontos. Tudni kell, hogy a hogy a szavak és mondatok milyen környezetben, milyen feltételezett szándékkal hangzottak el, kitől és kinek szólt a közlés.” (Péter, 2018)

1.3 A dolgozat célja és felépítése

A dolgozat célja a transzfer tanulás bemutatása és alkalmazása az NLP-ben. A dolgozat részletesen tárgyalja a transzfer tanulás fogalmát, előnyeit és gyakorlati alkalmazásait, különös tekintettel az NLP-re, vagyis a természetes nyelvi feldolgozásra.

Ezen kívül betekintést nyerünk a természetes nyelvi feldolgozás (NLP) történetébe és fejlődésébe, mely során megismerkedünk a terület legfontosabb mérföldköveivel és azzal, hogy hogyan jutottunk el a jelenlegi fejlett technológiákig. Az NLP főbb problémái és kihívásai is részletesen bemutatásra kerülnek, melyek közé tartozik például a többértelműség kezelése és a nyelvi változatosság.

A dolgozat áttekinti az NLP kulcsfontosságú technikáit és eljárásait, amelyek nélkülözhetetlenek a hatékony nyelvi feldolgozáshoz. Ezek közé tartoznak a szintaktikai és szemantikai elemzési módszerek, valamint a gépi fordítás és a nyelvi modellek fejlesztése.

A transzfer tanulás alkalmazása az NLP-ben külön fejezetben kerül tárgyalásra, ahol konkrét példák és esettanulmányok segítségével ismertetjük azokat a gyakorlati alkalmazásokat, amelyekkel a transzfer tanulás hatékonysága demonstrálható. Emellett megvizsgáljuk a nagy nyelvi modellek, mint például a GPT-3 hatásait és kihívásait az NLP területén.

A dolgozat további részeiben kitérünk az NLP alkalmazásaira különböző iparágakban, beleértve az egészségügyet, az ügyfélszolgálatot és a pénzügyi szektort. Ismertetjük azokat a technikai módszereket is, amelyek az NLP-ben használatosak, mint például a neurális hálók és a mélytanulási technikák.

A fejlett NLP technikák, a tokenizáció és szegmentálás, valamint az etikai kérdések és kihívások szintén fontos részét képezik a dolgozatnak. A korpuszok létrehozása és használata, a modellek fejlesztése is.

2. A TERMÉSZETES NYELVI FELDOLGOZÁS TÖRTÉNETE ÉS FEJLŐDÉSE

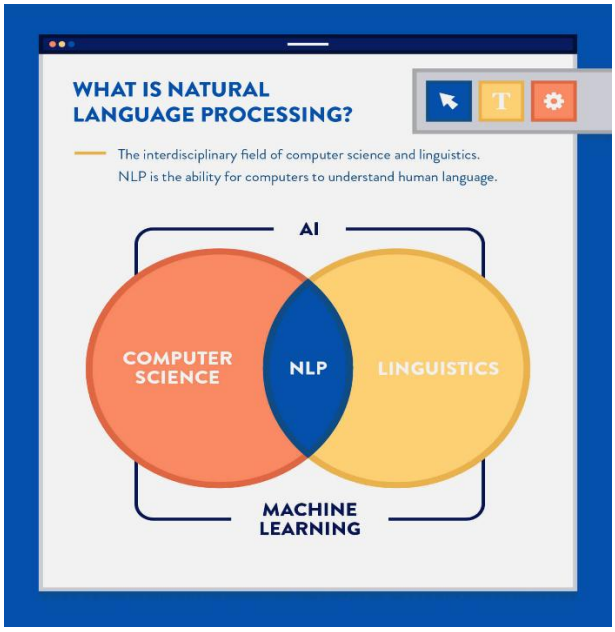
2.1 Korai módszerek: Szabályalapú rendszerek

A természetes nyelvi feldolgozás története során az első módszerek szabályalapú rendszerek voltak. Ezek a rendszerek kézzel írt szabályokon alapultak, amelyek a nyelvi struktúrák és minták leírására szolgáltak. Például a 1960-as években a SHRDLU nevű rendszer képes volt egyszerű parancsokat végrehajtani és válaszokat adni, de csak korlátozott kontextusban működött. Ezek a rendszerek alapvetően korlátozottak voltak, mivel nem tudtak megbirkózni a nyelv változékonyságával és a kontextusfüggő jelentéssel (Durai, 2021)

2.2 Statisztikai módszerek megjelenése

A statisztikai módszerek megjelenése jelentős előrelépést hozott az NLP területén. Az 1990-es években a statisztikai gépi fordítási rendszerek, mint például az IBM által fejlesztett modell, képesek voltak nagy mennyiségű kétnyelvű szöveg alapján előrejelzéseket készíteni a fordításokra. Ezek a modellek a valószínűségi eloszlásokat használták a nyelvi minták tanulására és alkalmazására, ami jelentős előrelépést jelentett a szabályalapú megközelítésekhez képest. Például a statisztikai módszerek lehetővé tették, hogy a gépi fordítók jobban kezeljék a többértelmű szavakat, mivel képesek voltak figyelembe venni a kontextust (Shaikh J. , 2017)

2.3 Gépi tanulás és mélytanulás hatása az NLP-re



1. ábra Mi is az NLP

(Zoltán, 2024)

A gépi tanulás és a mélytanulás új lehetőségeket nyitott az NLP-ben. A gépi tanulás algoritmusok, mint a Support Vector Machines (SVM) és a Random Forest, képesek voltak nagy mennyiségű adatból tanulni és általánosítani a nyelvi mintákat. A mélytanulás még tovább vitte ezt a fejlődést a neurális hálózatok alkalmazásával. A Convolutional Neural Networks (CNN) és a Recurrent Neural Networks (RNN) alkalmazása lehetővé tette a bonyolult nyelvi minták hatékony feldolgozását. Például a Long Short-Term Memory (LSTM) hálózatok használata javította a szekvenciális adatok feldolgozását és előrejelzését. A Google BERT (Bidirectional Encoder Representations from Transformers) modellje, amelyet 2018-ban mutattak be, jelentős áttörést hozott azáltal, hogy a nyelvi kontextust mindkét irányban figyelembe veszi, és így pontosabb értelmezéseket tesz lehetővé (Jacob Devlin, 2019)

2.4 Jelenlegi trendek és irányzatok

2. ábra NLP Felhasználhatósága

(Zoltán, 2024)

Az NLP kutatása jelenleg több fontos területre összpontosít:

Nyelvi sokféleség és komplexitás

A világon több mint 7000 nyelv létezik, és az NLP-modelleknek képesnek kell lenniük különböző nyelvek megértésére és feldolgozására. Például a Facebook AI által fejlesztett XLM-R (Cross-lingual Language Model) több mint 100 nyelvet támogat, és képes többnyelvű szövegeket kezelni hatékonyan.

Kontextus megértése

A szavak jelentése a kontextustól függően változhat, így az NLP-modelleknek képesnek kell lenniük a kontextus figyelembevételére. A Transformer modellek, mint például a BERT és a GPT, különösen hatékonyak ebben a tekintetben, mivel képesek hosszú távú függőségeket figyelembe venni a szövegben.

Szintaktikai és szemantikai elemzés

Az NLP modelleknek képesnek kell lenniük a mondatok szintaktikai és szemantikai elemzésére, ami szükséges a mondatok jelentésének megértéséhez. Például a Constituency Parsing és Dependency Parsing technikák használata.

Érzelmek és pragmatika

Az emberi nyelv gyakran tartalmaz érzelmeket és pragmatikus információkat, amit az NLP-modelleknek is képesnek kell lenniük felismerni. Az érzelemfelismerő rendszerek például a közösségi média elemzésében hasznosak. A Twitter sentiment analízise során például a modellek képesek azonosítani a pozitív, negatív vagy semleges érzelmeket a felhasználói bejegyzésekben.

Adatminőség kihívásai

Az NLP-modellek hatékony betanításához nagy mennyiségű, jó minőségű adatra van szükség. Az adatok tisztítása és előkészítése időigényes és költséges feladat. A különböző forrásokból származó adatok integrálása és homogenizálása szintén kihívást jelenthet, például amikor különböző nyelvi dialektusokat kell egyesíteni egy egységes adatbázisban (Colin Raffel N. S., 2020)

3. AZ NLP FŐBB PROBLÉMÁI

3.1 Szarkazmus és irónia

Az ilyen mondatokat nehéz felismerni az NLP-rendszerekben, mivel meg kell érteniük a szövegkörnyezetet és a beszélő szándékát. Például az "Ez tényleg fantasztikus!" mondat szarkasztikus lehet, ha valaki egy hibás termékre reagál vele. A szarkazmus felismerésére fejlesztett modellek, mint a DeepMoji, amely mélytanulást alkalmaz, képesek bizonyos fokig azonosítani ezeket a finom nyelvi jelenségeket. A szarkazmus és irónia pontos felismerése azonban továbbra is jelentős kihívást jelent az NLP számára, mivel ezek a nyelvi elemek gyakran finom kontextuális jeleken alapulnak, amelyek nehezen formalizálhatók (Durai, 2021)

3.2 Több nyelv

Mivel világszerte több mint 7000 nyelvet beszélnek, ezért óriási probléma azokat hatékonyan feldolgozni és NLP rendszereket kifejleszteni. A többnyelvű NLP-rendszerek fejlesztéséhez a szintaxis, a szabályok és a kulturális jellemzők megértése szükséges. Például a Facebook AI által fejlesztett XLM-R (Cross-lingual Language Model) több mint 100 nyelvet támogat, és képes többnyelvű szövegeket kezelni hatékonyan. A különböző nyelvek sajátosságainak és kulturális kontextusainak figyelembevétele azonban további kihívásokat jelent. A többnyelvű modellek esetében fontos, hogy a modell képes legyen kezelni a nyelvek közötti különbségeket, mint például a szórendi eltéréseket és a kulturális jelentésárnyalatokat

3.4 Szleng kifejezések

Az szleng kifejezések olyan szavak vagy kifejezések, amelyek a szó szerinti jelentésüktől eltérőre utalnak. Például az "On fleek" kifejezés jelentése "tökéletes", de ezt a hagyományos szótárak nem tartalmazzák. Az NLP-modelleknek folyamatosan frissülniük kell, hogy lépést tartsanak az új szleng kifejezésekkel és azok jelentésével. A szleng felderítésére és feldolgozására specializált adatbázisok és modellek, mint az Urban Dictionary API, segítséget nyújtanak ezen a területen. Az NLP modellek számára a szleng felismerése és értelmezése különösen fontos a közösségi média tartalmak elemzésekor, ahol gyakran használnak informális nyelvezetet és új szleng kifejezéseket (Colin Raffel N. S., 2020)

4. ETIKAI KÉRDÉSEK ÉS KIHÍVÁSOK AZ NLP-BEN

4.1 Adatvédelem

Az adatgyűjtés és feldolgozás során figyelembe kell venni az adatvédelmi törvényeket és előírásokat, mint például az EU általános adatvédelmi rendeletét (GDPR). Az adatvédelmi szabályok betartása elengedhetetlen a felhasználók bizalmának megőrzéséhez és az adatok etikus kezeléséhez.

4.2 Elfogultság és diszkrimináció

Az NLP modellekben rejlő elfogultságok komoly etikai kihívást jelentenek. Az elfogultságok felismerése és minimalizálása kulcsfontosságú a tisztességes és megbízható modellek fejlesztéséhez.

Az alábbi Python kód egy előre betanított nyelvi modellt (konkrétan egy BERT modellt) használ arra, hogy azonosítsa és bemutassa az abban rejlő esetleges elfogultságokat.

A kód egy olyan feladatot hajt végre, ahol a modellnek ki kell egészítenie egy mondatot egy hiányzó szóval (MASK). A példában a mondat "The doctor was a [MASK]." Ennek a feladatnak a segítségével megfigyelhetjük, hogy a modell milyen szavakat javasol a hiányzó helyre. Ha a modell például túlnyomórészt férfi foglalkozásokat javasol, az utalhat arra, hogy az adatokon, amelyeken a modellt betanították, az orvosok túlnyomórészt férfiakként voltak reprezentálva, így a modell ezt az elfogultságot átveszi. Ezzel a módszerrel a fejlesztők felismerhetik és elemezhetik a modellben rejlő elfogultságokat, és lépéseket tehetnek azok csökkentésére, például az adatok diverzifikálásával vagy a modell architektúrájának módosításával.

A kód tehát egy egyszerű, de hatékony eszközt biztosít az NLP modellekben rejlő elfogultságok feltárására.

```
from transformers import pipeline
# Elfogultság detektálása egy előre betanított modellen
unmasker = pipeline('fill-mask', model='bert-base-uncased')
result = unmasker("The doctor was a [MASK].")
print(result)
```

3. ábra Unmasker

[{'score': 0.08274320513010025, 'token': 11067, 'token_str': 'genius', 'sequence': 'the doctor was a genius.'}, {'score': 0.07349821925163269, 'token': 7966, 'token_str': 'fool', 'sequence': 'the doctor was a fool.'}, {'score': 0.024674762040376663, 'token': 3460, 'token_str': 'doctor', 'sequence': 'the doctor was a doctor.'}, {'score': 0.023262616246938705, 'token': 4393, 'token_str': 'vampire', 'sequence': 'the doctor was a vampire.'}, {'score': 0.022237030789256096, 'token': 2658, 'token_str': 'professional', 'sequence': 'the doctor was a professional.'}]

4.3 Átláthatóság és magyarázhatóság

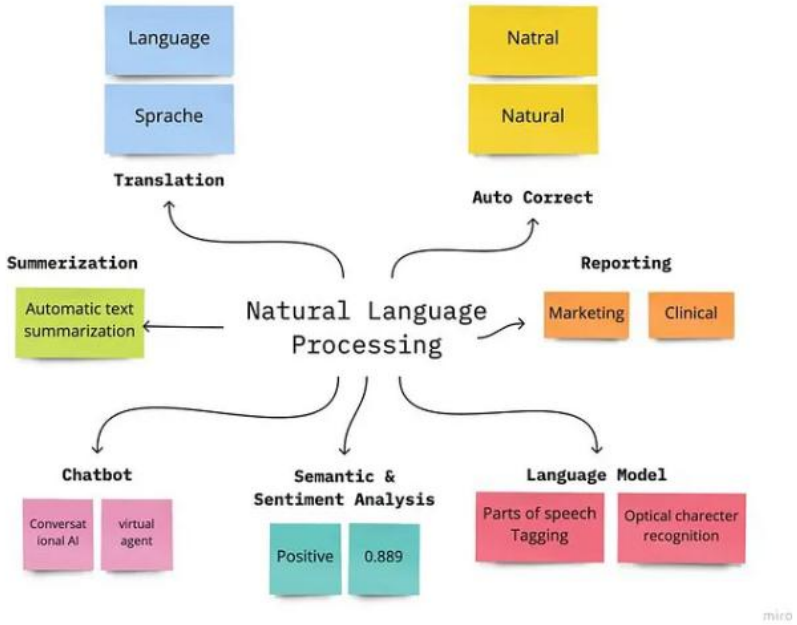
Az NLP modellek döntéseinek átláthatósága és magyarázhatósága kulcsfontosságú a felhasználói bizalom és a modellek megbízhatóságának biztosítása szempontjából. Az átláthatósági eszközök segítségével a felhasználók mélyebb betekintést nyerhetnek a modellek döntéshozatali folyamataiba, így könnyebben érthetővé válik, hogy egy adott válasz vagy javaslat miért született. Az átláthatóság növelésére irányuló technikák, mint például a Local Interpretable Model-agnostic Explanations (LIME) és a SHapley Additive exPlanations (SHAP), lehetővé teszik, hogy a modellek döntéseit lokális szinten magyarázzuk el, azaz egy-egy konkrét esethez kapcsolódóan.

A LIME módszer például segít a modellek interpretációjában, hogy kisebb, módosított bemeneti adatminták alapján készít magyarázatokat, és ezekből következtetéseket von le a modell eredeti döntéseinek okairól. Így a felhasználó érthetőbb és intuitívabb képet kap arról, milyen tényezők hatottak leginkább az adott döntésre (Marco Tulio Ribeiro, 2016)

A SHAP módszer a Shapley-értékekre épül, amelyek az egyes bemeneti változók hozzájárulását elemzik a modell kimeneti döntéséhez. Ez lehetővé teszi, hogy pontosabb, matematikailag megalapozott értékelést adjon az egyes bemenetek szerepéről és súlyáról az adott döntés kapcsán (Scott M. Lundberg, 2017)

Ezen technikák alkalmazása egyre elterjedtebb az NLP modellek magyarázhatóságának javítása érdekében, és az ilyen típusú megközelítések különösen fontosak a felhasználói elfogadás növelésében és a megbízhatóság fokozásában az üzleti alkalmazásokban.

5. NLP ALKALMAZÁSA KÜLÖNBÖZŐ IPARÁGAKBAN



Credits: Md Mahmud Hasan

4. ábra NLP Alkalmazása

(Singh, LLM Architectures Explained: NLP Fundamentals (Part 1), 2024)

5.1 Egészségügy

Az NLP alkalmazása az egészségügyben jelentős előrelépéseket hozott az elektronikus egészségügyi feljegyzések (EHR) feldolgozásában. Az EHR adatok strukturálatlan formában tartalmazzák a betegadatokat, így az NLP technikák segítenek ezek feldolgozásában és értelmezésében. Az NLP-alapú rendszerek képesek azonosítani és kinyerni fontos információkat, mint például a diagnózisokat, gyógyszereket és orvosi eljárásokat, ami hozzájárul a pontosabb és gyorsabb betegellátáshoz.

5.1.1 Betegadatok elemzése

Az NLP technológiák lehetővé teszik a betegadatok elemzését és a betegségek előrejelzését. Az NLP-alapú elemzések segíthetnek a korai felismerésben és a megelőző intézkedések meghozatalában, ami javítja a betegek egészségügyi kimenetelét.

5.1.2 Diagnosztikai rendszerek

Az NLP-t alkalmazó diagnosztikai rendszerek képesek orvosi szövegek és képek alapján diagnózisokat készíteni. Az ilyen rendszerek hozzájárulnak a diagnosztikai pontosság növeléséhez és az orvosi ellátás hatékonyságának javításához.

Fejlett Képalkotó és Mesterséges Intelligencia Alkalmazása az Agydaganatok Diagnosztikájában

Az agydaganatok kezelése gyakran műtéttel kezdődik, ahol a cél a daganatos szövetek minél nagyobb részének eltávolítása. Az eltávolított tumorminta elemzése segíthet a pontos diagnózis felállításában és a daganatos és egészséges szövetek közötti határvonal meghatározásában. Hagyományosan, ez az intraoperatív patológiai elemzés időigényes folyamat, amely során a mintát a patológus feldolgozza, megfesti és elemzi, míg a sebész és a beteg az eredményekre vár.

Az SRH-AI (Stimulált Raman-hisztológia és Mesterséges Intelligencia) technológia forradalmi újítás ezen a területen. Az SRH képalkotó technológia és egy mesterséges intelligencia algoritmus ötvözésével ez az eljárás képes kevesebb mint 3 perc alatt pontos diagnózist adni az agydaganatokról a műtét alatt. Az új megközelítés megbízhatóan különbséget tud tenni a tumorszövet és az egészséges agyszövet között.

Klinikai Vizsgálat és Pontosság

A technológia klinikai értékét egy közel 280 beteget bevonó vizsgálat során tesztelték. A szövetmintákat két részre osztották: az egyik részt a műtőben az SRH-AI technológiával, a másikat pedig a hagyományos patológiai módszerekkel elemezték. Az eredmények alapján az algoritmus 94,6%-os pontossággal diagnosztizálta az agydaganatokat, szemben a hagyományos patológia 93,9%-os pontosságával. Érdekes megfigyelés volt, hogy az algoritmus és a patológusok kölcsönösen javították egymás diagnosztikai pontosságát a nehezebb esetekben, rámutatva az MI technológia és az emberi szakértelem kombinálásának előnyeire.

Mesterséges Intelligencia és Mélytanulás az Orvosi Képfeldolgozásban

A mesterséges intelligencia alkalmazása az orvostudományban egyre elterjedtebbé válik, különösen a mélytanulás révén. A mélytanulási algoritmusok, mint a konvolúciós neurális hálózatok, képesek komplex mintázatok felismerésére nagy mennyiségű adat elemzése alapján. Az SRH-AI algoritmus betanításához 415 beteg több mint 2,5 millió tumorszöveti képét használták fel, amelyek lefedték az agydaganatok tíz leggyakoribb típusát és három egészséges szövettípust. A képzés során a kutatók optimalizálták a képek méretét és felbontását, hogy az algoritmus a legpontosabban tudja megkülönböztetni a tumoros, nem tumoros és nem diagnosztikus szöveteket.

A Képalkotó Technológia és MI Jövője a Műtéti Gyakorlatban

Az SRH technológia, amely a mikroszkópia egy speciális formája, lehetőséget biztosít a friss szövetminták valós idejű vizualizációjára a műtőben. Az SRH olyan "festést" eredményez, amely hasonló a patológusok által alkalmazott módszerekhez, és a Michigani Egyetem sebészeti csapatai már sikeresen használják agydaganatok és fej-nyaki rákok esetén. Az MI-vel való kombinálás révén ez a technológia lehetőséget ad a gyorsabb és pontosabb intraoperatív döntéshozatalra, növelve a műtétek sikerességét és csökkentve a kockázatokat.

Ez a fejlett technológia potenciálisan forradalmasíthatja az intraoperatív patológiát, és példaként szolgál arra, hogy a mesterséges intelligencia hogyan segítheti az orvosi diagnosztika területén dolgozó szakembereket a még nagyobb pontosság és gyorsaság elérésében. A patológusok és a mesterséges intelligencia közös munkája olyan szinergiát eredményezhet, amely hozzájárul a betegek jobb ellátásához és a diagnosztikai hibák minimalizálásához. (Staff, 2020)

5.2 Pénzügyi szektor

5.2.1 Fraud detektálás

Az NLP technikák jelentős szerepet játszanak a pénzügyi csalások felismerésében és megelőzésében. Az NLP-alapú rendszerek képesek elemezni a tranzakciós adatokat és azonosítani a gyanús tevékenységeket. Ezek a rendszerek valós időben figyelik és elemzik a pénzügyi tranzakciókat, lehetővé téve a csalások gyors felismerését és megelőzését.

5.2.2 Hitelképesség-elemzés

Az NLP segít a hitelképesség-elemzésben azáltal, hogy elemezi a hitelkérelmeket és a pénzügyi dokumentumokat. Az NLP-alapú rendszerek képesek értékelni a hitelkérelmezők pénzügyi helyzetét és kockázati profilját, ami hozzájárul a hitelkérelmek gyors és pontos elbírálásához.

5.2.3 Ügyfélszolgálati chatbotok

Az ügyfélszolgálati chatbotok az NLP technológiák egyik legnépszerűbb alkalmazási területe. Az NLP-alapú chatbotok képesek természetes nyelven kommunikálni az ügyfelekkel, megértik a kérdéseket és releváns válaszokat adnak. Ezek a chatbotok csökkentik az ügyfélszolgálati terhelést és javítják az ügyfélélményt.

5.3 Jog

5.3.1 Jogszabályok és szerződések elemzése

Az NLP technikák segítenek a jogi dokumentumok, például jogszabályok és szerződések elemzésében. Az NLP-alapú rendszerek képesek automatikusan kinyerni a fontos információkat és azonosítani a jogi kifejezéseket, ami hozzájárul a jogi dokumentumok gyorsabb és pontosabb feldolgozásához.

5.3.2 Bírósági döntések előrejelzése

Az NLP alkalmazása a bírósági döntések előrejelzésében lehetővé teszi a jogi precedensek és bírósági döntések elemzését. Az NLP-alapú rendszerek képesek azonosítani a bírósági döntések mintáit és előre jelezni a jövőbeli döntéseket, ami segíthet a jogi stratégiák kidolgozásában.

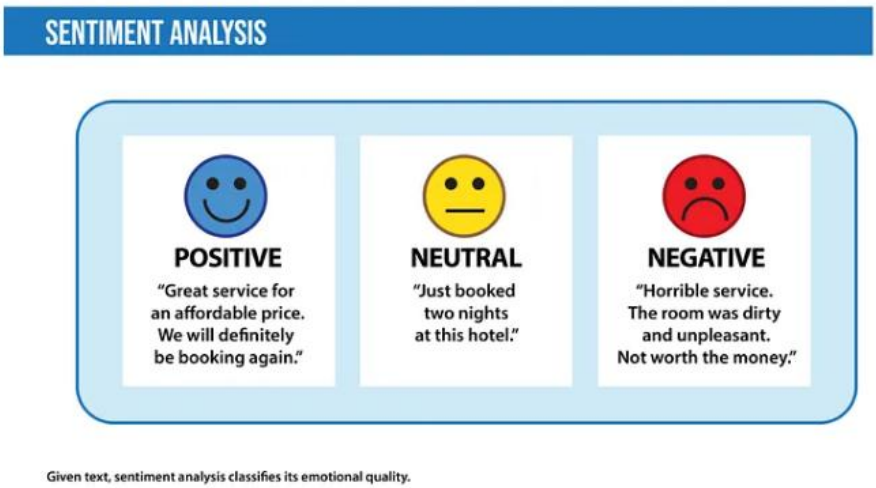
5.4 Marketing

5.4.1 Sentimental analysis

A természetes nyelvfeldolgozás (NLP) technikáival végzett sentimental analysis fontos eszköz a modern marketing számára, különösen a közösségi média és az ügyfélértékelések világában. Az NLP-alapú rendszerek képesek azonosítani a szövegekben megjelenő érzelmeket, amelyek lehetnek pozitívak, negatívak vagy semlegesek. Ez az információ különösen hasznos a vállalatok számára, mivel így jobban megérthetik ügyfeleik véleményét, igényeit és elégedettségi szintjét.

Az NLP és sentimental analysis révén a vállalatok képesek gyorsan és hatékonyan reagálni az ügyfelek visszajelzéseire, akár valós időben is. Például, ha egy termékről vagy szolgáltatásról negatív visszajelzés érkezik a közösségi médiában, az automatikus érzelemelemzés azonnal figyelmeztetheti az ügyfélszolgálatot, hogy megfelelő lépéseket tegyenek. Ugyanakkor a pozitív visszajelzések elemzése segíthet az erősségek kiemelésében és a marketingstratégiák optimalizálásában.

A sentimental analysis egyik gyakran alkalmazott eszköze a BERT (Bidirectional Encoder Representations from Transformers) modell, amely képes figyelembe venni a szöveg kontextusát mindkét irányból, így pontosabb előrejelzéseket nyújt az érzelmek azonosításában.



5. ábra Sentiment analysis

(Singh, LLM Architectures Explained: NLP Fundamentals (Part 1), 2024)

5.4.1.1 A Reddit Sentimental Analysis Alkalmazása

A Reddit felhasználói közösségeken alapuló struktúrája egyedülálló lehetőséget biztosít a szentimentelemzés számára. A Reddit felhasználói gyakran részt vesznek mélyebb, részletes beszélgetésekben, így az ott megjelenő érzelmi mintázatok és hangulatok értékes adatforrást jelenthetnek a vállalatok számára. A szentimentelemzés során a vállalatok célzottan vizsgálhatják azokat a subreddit közösségeket, amelyek relevánsak számukra, például egy-egy termék, szolgáltatás vagy iparág szempontjából. Ennek segítségével azonosíthatók a felhasználói elégedettség, csalódottság vagy éppen pozitív reakciók, amelyek befolyásolhatják a marketingstratégiát.

A Reddit szentimentelemzés különösen hasznos a következő területeken:

- **Versenyelemzés**: A vállalatok összehasonlíthatják saját termékeik vagy szolgáltatásaik iránti érdeklődést és elkötelezettséget a versenytársaikéval. Ez segít megérteni, hogy a fogyasztók hogyan érzékelik az egyes márkákat, és milyen változtatásokat várnak.
- **Termékfejlesztés és fejlesztési visszajelzések**: A Reddit hozzászólások gyakran tartalmaznak hasznos visszajelzéseket, amelyeket a vállalatok felhasználhatnak a termékek vagy szolgáltatások fejlesztésére. A szentimentelemzés segíthet azonosítani azokat a jellemzőket, amelyekkel a felhasználók elégedettek, illetve azokat, amelyek javításra szorulnak.

- **Reputációmenedzsment**: A Reddit lehetőséget biztosít arra, hogy valós idejű képet kapjunk a márka iránti közvéleményről. A negatív és pozitív visszajelzések elemzése hozzájárulhat a gyors reakciókhoz, ezáltal javítva a vállalat hírnevét és minimalizálva a lehetséges károkat.

5.4.1.2 A Szentimentelemzés Technikái Reddit Tartalmakra

A Reddit bejegyzések elemzéséhez használt szentimentelemző technikák közé tartozik a gépi tanulási és mélytanulási algoritmusok alkalmazása, amelyek képesek felismerni a szöveges tartalmak érzelmi tónusát. A Reddit tartalmak elemzéséhez különösen fontos figyelembe venni a platform specifikus jellemzőit, például az irónia, a szarkazmus és a humor gyakori jelenlétét, amelyek befolyásolhatják a felhasználók érzelmi kifejezéseit.

Az egyik leghatékonyabb eszköz a BERT (Bidirectional Encoder Representations from Transformers) modell, amely a kontextus figyelembevételével elemzi a szöveget, lehetővé téve a szarkazmus és az összetettebb érzelmek pontosabb felismerését. A BERT alapú szentimentelemző modellek képesek különbséget tenni az egyértelmű és összetett érzelmi kifejezések között, amelyek jellemzőek a Reddit bejegyzésekben.

```python
import torch
from transformers import BertForSequenceClassification, BertTokenizer

# Modell elérési útjának meghatározása
model_path = "./saved_model"

# Modell betöltése
model = BertForSequenceClassification.from_pretrained(model_path)

# Tokenizátor betöltése
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")

# Teszt bemenet
test_sentence = "Macko papa love his little bear"

# Input előkészítése és predikciók végrehajtása
inputs = tokenizer(test_sentence, return_tensors="pt")
outputs = model(**inputs)
logits = outputs.logits

# Logitok Softmax transzformációja
probabilities = torch.nn.functional.softmax(logits, dim=1)

# Predikciók feldolgozása és kimenet megjelenítése
predicted_class = torch.argmax(probabilities, dim=1).item()
class_labels = ["negative", "neutral", "positive"]  # osztálycímke lista

predicted_label = class_labels[predicted_class]

print(f"Predicted class: {predicted_label}")
```

```
Predicted class: positive
```

6. ábra Saját munka - Reddit Sentimental Analysis Házidolgozat Rendszerfejlesztés

A hagyományos senitmetal analysis főként a pozitív, negatív és semleges érzelmek azonosítására koncentrál, azonban egyre nagyobb az igény a mélyebb és részletesebb érzelemfelismerésre. A többdimenziós érzelemelemzés lehetőséget nyújt arra, hogy a modellek olyan érzelmeket is felismerjenek, mint a boldogság, meglepetés, szomorúság, harag, félelem vagy undor. A különböző érzelmi kategóriák változása segíthet a trendek felismerésében, például az ügyfelek érdeklődésének csökkenése, ha egy termékkel kapcsolatban növekvő mértékben tapasztalható szomorúság vagy csalódottság.

6. NLP A GYAKORLATBAN

6.1 Nyílt forráskódú eszközök és könyvtárak

Az alábbiakban bemutatok néhány népszerű nyílt forráskódú eszközt és könyvtárat, amelyeket az NLP-ben széles körben használnak:

Hugging Face: Az egyik legnépszerűbb NLP könyvtár, amely előre betanított modelleket és eszközöket kínál a szövegfeldolgozásra és a modell finomhangolására.

SpaCy: Egy gyors és hatékony NLP könyvtár, amely különösen hasznos a szöveg előfeldolgozásában és az NER feladatokban.

NLTK: Az egyik legrégebbi és legnépszerűbb NLP könyvtár, amely számos eszközt és erőforrást kínál a nyelvi feldolgozáshoz.

A lentebb lévő kód a természetes nyelvfeldolgozás (NLP) területén széles körben használt SpaCy könyvtár egy egyszerű alkalmazását mutatja be. A SpaCy egy gyors és hatékony eszköz, amely számos funkciót kínál a szövegek feldolgozására, például a tokenizálást, a részbeszó-címkézést és a név-entitás felismerést. A kódban először betöltjük az angol nyelvű kis méretű modellt ("en_core_web_sm"), majd egy minta mondatot adunk meg. A SpaCy segítségével elemezzük a mondatot, és a doc.ents attribútummal hozzáférünk a benne található név-entitásokhoz (pl. személynevek, szervezetek, helyek). A kód végül kiírja az azonosított entitásokat és azok típusát. Ez a példa jól szemlélteti, hogy a SpaCy segítségével milyen egyszerűen végezhetünk alapvető NLP feladatokat, mint például a szövegben szereplő fontos információk kinyerése.
A Hugging Face és az NLTK mellett a SpaCy a legnépszerűbb nyílt forráskódú NLP könyvtárak közé tartozik, amelyeket számos kutató és fejlesztő használ világszerte. Ezek a könyvtárak számos előre betanított modellt és eszközt kínálnak, amelyek jelentősen megkönnyítik az NLP projektek megvalósítását.

```python
import spacy
# SpaCy betöltése
nlp = spacy.load("en_core_web_sm")
# Szöveg elemzése
text = "Apple is looking at buying U.K. startup for $1 billion."
doc = nlp(text)
# Név entitás felismerés
for ent in doc.ents:
 print(ent.text, ent.label_)
```

7. ábra SpaCy

Apple ORG, U.K. GPE, $1 billion MONEY

7. JÖVŐBELI IRÁNYOK ÉS KUTATÁSI LEHETŐŐSÉGEK

Ahogy a természetes nyelvfeldolgozás (NLP) technológiai fejlődése új távlatokat nyit, egyre nagyobb hangsúlyt kap a modellek társadalmi, etikai és környezeti hatása, valamint az új technológiai trendek és megközelítések felfedezése. A jövőbeli kutatások és fejlesztések célja az NLP modellek intelligenciájának, hatékonyságának és felelősségteljes alkalmazásának növelése.

7.1 Etikai és társadalmi szempontok figyelembevétele

Az NLP-modellek fejlesztésénél az etikai és társadalmi felelősség kulcsfontosságú. Az NLP rendszerek széleskörű elterjedése és alkalmazása csak akkor valósulhat meg, ha a modellek tiszteletben tartják az emberi jogokat, minimalizálják az elfogultságot, és átlátható működést biztosítanak. Az NLP modellek, különösen a nagy nyelvi modellek, mint a BERT és GPT, képesek tanulni az emberek közötti kommunikációs mintákból, ugyanakkor fennáll annak a veszélye, hogy a torzított vagy nem reprezentatív adatok beépülnek a modellek működésébe, és ezek az elfogultságok kihatással lehetnek a modell eredményeire. Például egyes modellek hajlamosak nemi és faji sztereotípiákat tükrözni, ami veszélyezteti az objektív döntéshozatalt különböző alkalmazásokban.

A jövőbeli kutatások egyik fő célja az ilyen jellegű elfogultságok minimalizálása, amelyet az adathalmazok kiegyensúlyozásával és folyamatos monitorozásával érhetünk el. Az NLP-modellek átláthatóságának javítása szintén központi kérdés. Olyan technikák kidolgozása szükséges, amelyek megmutatják, hogy a modellek milyen adatok és folyamatok alapján döntenek, ezáltal biztosítva a felhasználók számára, hogy jobban megértsék a modellek működését és korlátait (Emily M. Bender, 2021).

7.2 Méretek és erőforrások összefüggése

Az NLP-modellek, különösen a mélytanulási technikákkal működő nagy nyelvi modellek jelentős számítási erőforrást igényelnek, ami környezeti és gazdasági kihívásokat vet fel. A modellek betanítása nagy mennyiségű energiát igényel, és komoly karbonlábnyommal járhat, különösen akkor, ha gyakori újratréningezésre van szükség. E problémák kezelésére a kutatók egyre inkább olyan megközelítések kidolgozására törekszenek, amelyek csökkentik az energiafogyasztást és a költségeket, ugyanakkor megőrzik a modellek hatékonyságát.

A méret optimalizálására szolgáló technikák, mint a modell prúnálás (Ez a technika a modellben található "felesleges" elemek (például bizonyos neurális kapcsolatokat) eltávolítását jelenti. Így a modell kisebb lesz, és kevesebb számítási erőforrást igényel, de a teljesítménye alig változik.) és a tudásátvitel (knowledge distillation - Ezzel a módszerrel a nagy, bonyolult modell tudását egy kisebb modellre átviszik. Ez a kisebb modell hasonló teljesítményt nyújt, de jóval kevesebb erőforrást igényel.), hatékonyan csökkenthetik a modellek számítási erőforrásigényét anélkül, hogy ez jelentős teljesítményveszteséggel járna. Ezenkívül a modellek adaptív tanulása (csak akkor tanítja újra a modellt, ha ez feltétlenül szükséges, például amikor új adatok állnak rendelkezésre vagy a környezet megváltozik. Ez a megközelítés csökkenti az energiafelhasználást és a költségeket, mivel elkerüli a felesleges újratanításokat.) lehetőséget nyújt a modellek újratréningezésére csak akkor, ha ez szükséges, csökkentve az energiafogyasztást és a költségeket Az ilyen optimalizációk nemcsak környezetbarátabb megoldásokat eredményezhetnek, hanem a kisebb szervezetek számára is lehetővé teszik a csúcstechnológiás NLP-modellek használatát. (Emma Strubell, 2019)

7.3 NLP Algoritmusok Fejlődése

Az NLP algoritmusok folyamatos fejlődése új lehetőségeket nyit az ember-gép kommunikáció terén. Az algoritmusok fejlesztése során a legnagyobb előrelépés a mélytanulási technikák, különösen a rekurrens neurális hálózatok (RNNs) és a transformer alapú modellek, például a BERT és GPT területén történt. Ezek az algoritmusok képesek nagy mennyiségű adatból tanulni, és komplex mintázatokat felismerni, lehetővé téve a gépek számára, hogy az emberi nyelvet természetes módon értelmezzék és használják.

Az NLP algoritmusok terén jelentős újítások történtek a pre-tréningelt nyelvi modellek alkalmazásával, amelyek már hatalmas szövegmennyiségen tanultak, és így képesek megérteni a kontextuális jelentéseket. Az ilyen modellek lehetővé teszik, hogy az NLP alkalmazások magasabb szintű teljesítményt érjenek el különböző nyelvi megértési feladatokban. A jövőben a kutatások célja, hogy a modellek még kifinomultabb kontextuális és érzelmi megértésre tegyenek szert, ezzel lehetővé téve, hogy jobban felismerjék az implicit szándékokat és érzelmeket is a felhasználói interakciók során (ColdSceptical, 2024)

7.4 Az NLP jövőbeli irányai: Kontextuális Megértés és Érzelemfelismerés

A kontextuális megértés fejlesztése a jövő egyik kulcsfontosságú területe az NLP-ben. A jelenlegi algoritmusok egyre inkább képesek a szöveg kontextusának mélyebb megértésére, ami azt jelenti, hogy nem csupán az egyes szavak vagy kifejezések jelentését értelmezik, hanem azt is, hogy ezek hogyan kapcsolódnak egymáshoz és az adott helyzethez. Ez lehetővé teszi, hogy a gépi rendszerek pontosabb és személyre szabottabb válaszokat adjanak, ami különösen fontos a közösségi médiában vagy a személyre szabott ügyfélszolgálati alkalmazásokban.

Az érzelemfelismerés szintén egyre nagyobb figyelmet kap, mivel a gépek képessége az emberi érzelmek azonosítására alapvető jelentőségű a felhasználói élmény javításában. Az NLP modellek fejlesztése során egyre fontosabbá válik, hogy a rendszerek képesek legyenek érzékelni a felhasználók hangulatát és ennek megfelelően reagálni, ezáltal növelve az empatikus kapcsolatot a felhasználók és a gépi rendszerek között. Az érzelemfelismerés kiemelten hasznos lehet az ügyfélszolgálatban, terápiás alkalmazásokban és más interaktív platformokon. (ColdSceptical, 2024)

7.5 Multimodális NLP és kontextus alapú elemzés

A multimodális NLP technológiák lehetőséget nyújtanak a szöveges és nem szöveges adatok együttes elemzésére, például képek, hangok és videók alapján. Az ilyen multimodális megközelítések különösen hasznosak a kontextus megértésében, hiszen a felhasználói érzelmek és hangulatok értékeléséhez nem elegendő csupán a szavakat figyelembe venni, hanem a vizuális és auditív jeleket is. A multimodális elemzési technikák hozzájárulhatnak a felhasználói élmény javításához, különösen a közösségi média elemzésében, ahol a felhasználók érzelmi reakciói fontos információkat nyújtanak a márkák és termékek fogadtatásáról.

A jövőbeli kutatások célja olyan integrált modellek létrehozása, amelyek képesek különböző típusú adatok együttes feldolgozására és a szöveges tartalmak kontextusba helyezésére. Ez a megközelítés nemcsak az NLP modellek pontosságát növeli, hanem lehetőséget biztosít komplexebb elemzésekre is, például a felhasználói élmény személyre szabásában és az interakciók dinamikusabbá tételében. (Binxuan Huang, 2019)

8. AZ NLP KULCSFONTOSSÁGÚ TECHNIKÁI ÉS ELJÁRÁSAI

8.1 Hagyományos és Mélytanulás-alapú NLP Technológiák

Az NLP (természetes nyelvfeldolgozás) technikák számos megközelítést alkalmaznak a szöveges adatok elemzésére, beleértve a hagyományos gépi tanulási algoritmusokat és a modern mélytanulási modelleket. Ezek a technikák lehetővé teszik különböző nyelvi feladatok megoldását, például érzelemelemzést, spamszűrést, gépi fordítást, szövegosztályozást és szövegösszefoglalást. Az alábbiakban áttekintjük a legfontosabb hagyományos és mélytanulás-alapú megközelítéseket.

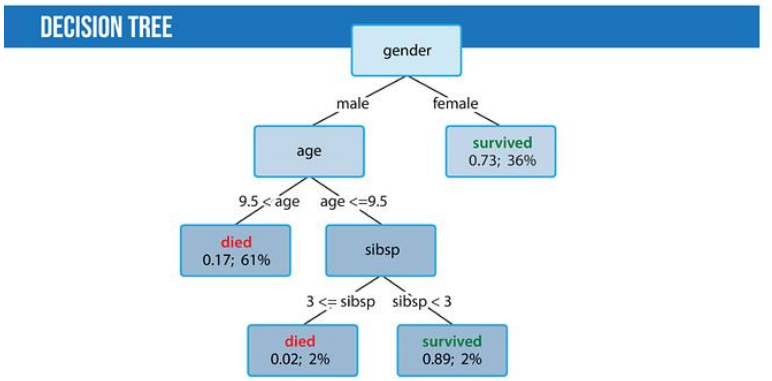
8.1.1 Hagyományos Gépi Tanulási Technikák NLP Feladatokra

A hagyományos gépi tanulási algoritmusok strukturáltabbak, és általában kisebb számítási kapacitást igényelnek, mint a mélytanulási modellek. Ezen algoritmusok alkalmazása a nyelvfeldolgozás terén az alábbiak szerint történik:

Logisztikus Regresszió: Felügyelt tanulási algoritmus, amely az események bekövetkezésének valószínűségét előrejelzi adott bemeneti változók alapján. Az NLP-ben gyakran alkalmazzák érzelemelemzésre, spamszűrésre és toxicitás osztályozására.

Naive Bayes: A Bayes-tétel alapján működő, felügyelt osztályozási algoritmus, amely az egyes szavak függetlenségét feltételezi. A Naive Bayes modell hatékonyan alkalmazható például spamszűrésre és hibakeresésre szoftverkódokban.

Döntési Fák: Ezek az algoritmusok jellemzők szerinti felosztásokkal osztályozzák az adatokat, miközben az információnyereséget maximalizálják. Az NLP terén olyan feladatokban használják, ahol a szöveg különböző jellemzői alapján történő osztályozásra van szükség.

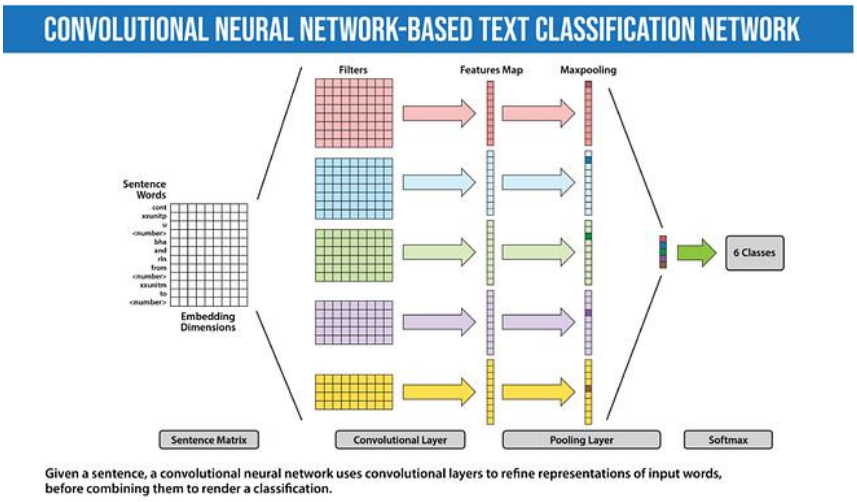


8. ábra Döntési fa *(deeplearning.ai, 2023)*

8.1.2 Mélytanulás-alapú NLP Technológiák

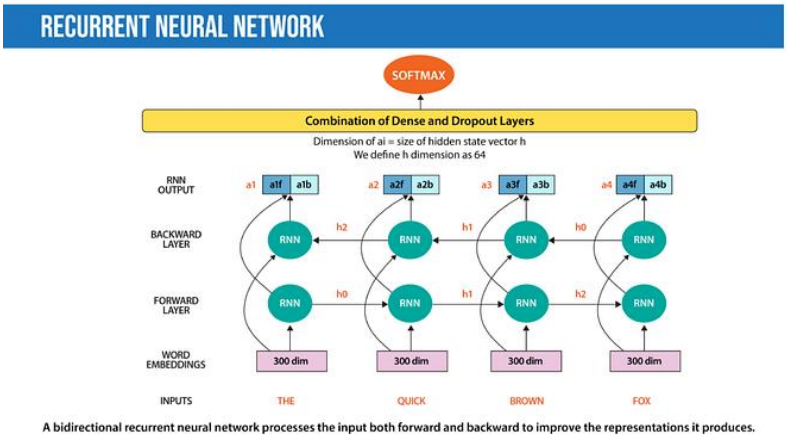
A mélytanulási technológiák nagy előnye, hogy képesek komplex nyelvi mintázatokat és szekvenciákat felismerni, ezáltal jelentősen hozzájárulnak a nyelvi adatok pontosabb elemzéséhez és feldolgozásához. Az alábbiakban bemutatjuk a legfontosabb mélytanulási technikákat:

Konvolúciós Neurális Hálózatok (CNN): A CNN-eket kezdetben képfeldolgozásra fejlesztették, de az NLP-ben is hatékonyan használhatók, például mondatok osztályozásában. Itt a szöveget szómátrixként reprezentálják, és a CNN kiemeli a releváns mintákat.



9. ábra Konvulúciós Neurális Hálózatok (CNN) *(deeplearning.ai, 2023)*

Rekurzív Neurális Hálózatok (RNN): Az RNN-ek képesek megőrizni az előző állapotok információit és összekapcsolni őket a jelenlegi feladattal. Az NLP-ben az LSTM (Long Short-Term Memory) és a GRU (Gated Recurrent Unit) típusú RNN-eket széles körben használják, mivel képesek hosszú távú függőségek kezelésére, például a szövegkörnyezet figyelembevételére.

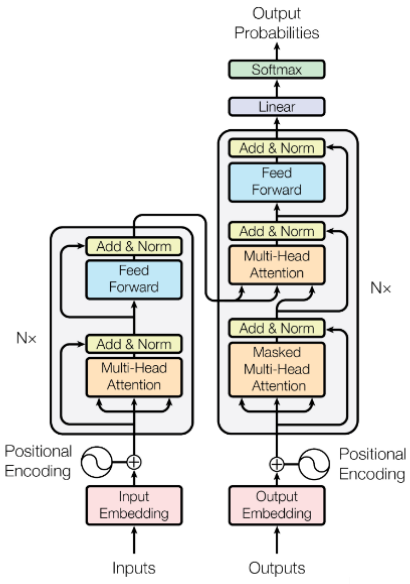


10. ábra Rekurzív Neurális Hálózatok (RNN) *(deeplearning.ai, 2023)*

Autoenkóderek: Ezek az önálló kódolási és dekódolási hálózatok a bemeneti adatokat alacsonyabb dimenziójú reprezentációba tömörítik, majd rekonstruálják az eredeti bemenetet. Az NLP-ben gyakran használják dimenziócsökkentésre és jellemzők kivonására.

Encoder-Decoder Szekvencia-Szekvencia Modellek: Ez az architektúra lehetőséget biztosít a szövegek tartalmának átalakítására, például gépi fordítás vagy szövegösszefoglalás esetén. Az encoder összegyűjti a szöveges információkat egy tömörített reprezentációba, míg a decoder ezt az információt a kívánt kimenet generálásához használja fel.

Transzformerek: A transformer modellek új irányt képviselnek az NLP-ben, amely az önfigyelési mechanizmuson (self-attention) alapul. Ez a mechanizmus lehetővé teszi, hogy a modell minden egyes szóra figyelmet fordítson, így egyszerre képes kezelni a teljes szöveget. Az olyan modellek, mint a BERT és GPT, transzformer alapúak, és a nyelvi feldolgozásban új szabványokat állítottak fel.



11. ábra Transformer (Jung, 2024)

- **Önfigyelési Mechanizmus (Self-Attention):** Az önfigyelés lehetővé teszi, hogy a modell figyelembe vegye a szavak közötti kapcsolatokat, ezáltal pontosabban értelmezze a szövegeket.
- **Többfejű Figyelem (Multi-Head Attention):** Ez a technika több szempont alapján figyeli a szöveget, így különböző nézőpontokból értelmezheti a szavak közötti összefüggéseket.

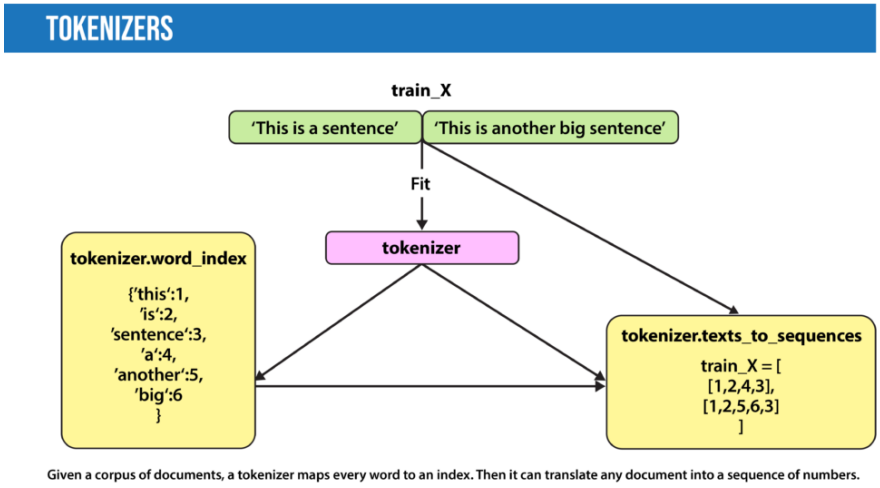
- **Pozíciós Kódolás:** Mivel a transzformer modellek nem érzékelik a szavak sorrendjét, pozíciós kódolás segítségével biztosítják, hogy a modell figyelembe vegye a szöveg szerkezeti sorrendjét.

Ezek a deeplearning technikák forradalmasították az NLP-t, lehetővé, hogy mélyebb kontextusban és hosszabb távú függőségekkel rendelkező szövegeket is feldoglozzanak. (Singh, LLM Architectures Explained: NLP Fundamentals (Part 1), 2024)

8.1 Tokenizáció

8.1.1 Tokenizáció fogalma és jelentősége

A tokenizáció az NLP egyik alapvető folyamata, amely során a szöveget kisebb egységekre, tokenekre bontjuk. Ezek a tokenek lehetnek szavak, szótöredékek vagy karakterek. A tokenizáció a szövegfeldolgozás első lépése, és kulcsfontosságú a gépi tanulási modellek számára, mivel segít a szöveg struktúrájának és jelentésének azonosításában.



12. ábra Tokenizáció *(deeplearning.ai, 2023)*

8.1.2 Tokenizáció típusai

- **Szavas tokenizáció**: A szöveg szavakra bontása. Például a "Hello, world!" mondatból ["Hello", ",", "world", "!"] lesz.
- **Szótöredék alapú tokenizáció**: A szavakat kisebb egységekre bontja, ami hasznos lehet ritka szavak esetén. Például a "unhappiness" szó "un", "happiness" tokenekre bontható.
- **Karakter alapú tokenizáció**: A szöveg karakterekre bontása. Például a "Hello" szó ["H", "e", "l", "l", "o"] tokenekre bontható.

8.1.3 Tokenizáció gyakorlati alkalmazása

A tokenizációt számos NLP könyvtár és eszköz támogatja, mint például a Hugging Face Tokenizers könyvtára, amely hatékony eszközöket kínál a szöveg tokenizálására. A lentebb lévő Python kód egy konkrét példát mutat be a természetes nyelvfeldolgozás (NLP) területén gyakran használt tokenizáció folyamatára, amelyet a Hugging Face Tokenizers könyvtár segítségével valósít meg. A tokenizáció lényege, hogy a szöveget kisebb, értelmes egységekre, tokenekre bontjuk, amelyek lehetnek szavak, karakterek vagy kifejezések. Ez az első és alapvető lépése a legtöbb NLP feladatnak. A kódban először betöltünk egy előre betanított BERT modelltől származó tokenizálót, majd egy egyszerű mondatot adunk meg bemenetként. A tokenizer.tokenize() függvény segítségével ezt a mondatot tokenekre bontjuk, és a kapott tokeneket egy listában jelenítjük meg. Így a kód szemlélteti, hogy egy népszerű NLP könyvtár segítségével milyen egyszerűen végezhetünk tokenizálást, ami számos további NLP feladat alapja lehet, mint például a szövegbesorolás, a gépi fordítás vagy a kérdés-felelet rendszerek.

```
from transformers import BertTokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
text = "Hello, world!"
tokens = tokenizer.tokenize(text)
print(tokens)
```

13. ábra Bert Tokenizer

['hello', ',', 'world', '!']

8.1.4 Tokenizáció és Szegmentálás

Fejlett tokenizációs módszerek

A fejlett tokenizációs módszerek, mint a Byte-Pair Encoding (BPE), a WordPiece és a SentencePiece, lehetővé teszik a szövegek hatékonyabb feldolgozását és kisebb szólisták létrehozását. Ezek a módszerek különösen hasznosak a ritka szavak kezelésében és a modell teljesítményének javításában.

A lentebb lévő Python kód egy fejlett tokenizációs módszer, a SentencePiece alkalmazását mutatja be. A hagyományos tokenizáció során a szöveget egyszerűen szavakra bontjuk. A SentencePiece azonban intelligens módon, statisztikai elemzés alapján határozza meg, hogy mely karakterkombinációkat kezeljünk egyetlen tokenként. Ez különösen hasznos olyan nyelvek esetében, ahol a szóhatárok nem mindig egyértelműek, vagy ahol sok ritka szó fordul elő.

A kódban először betanítunk egy SentencePiece modellt egy adott szöveges adathalmazon, majd ezt a modellt használjuk egy új szöveg tokenizálására. A modell létrehozásakor megadhatjuk a kívánt szójegyzék méretét, ami befolyásolja a tokenek finomságát. A SentencePiece előnye, hogy képes kezelni ismeretlen szavakat is, és kisebb szójegyzéket eredményez, ami hatékonyabbá teszi a gépi tanulási modelleket.*python*

```python
import sentencepiece as spm
# Train SentencePiece model
spm.SentencePieceTrainer.train(input='text.txt', model_prefix='spm', vocab_size=200)
# Load SentencePiece model
sp = spm.SentencePieceProcessor(model_file='spm.model')
# Tokenize text
text = "This is an example sentence."
tokens = sp.encode_as_pieces(text)
print(tokens)
```

14. ábra Sentence Piece modell

['▁', 'T', 'h', 'i', 's', '▁', 'i', 's', '▁a', 'n', '▁', 'e', 'x', 'a', 'mp', 'l', 'e', '▁s', 'en', 't', 'en', 'c', 'e', '.']

Nyelvspecifikus tokenizáció

A különböző nyelvek sajátosságai kihívásokat jelentenek a tokenizáció számára. Például a japán és kínai nyelvek nem használnak szóközöket a szavak elválasztására, míg a magyar nyelv komplex morfológiája kihívást jelenthet a tokenizáció során.

A lenti Python kód egy konkrét példát mutat be a természetes nyelvfeldolgozás (NLP) területén gyakran előforduló kihívásra, a nyelvspecifikus tokenizációra. A tokenizáció lényege, hogy a szöveget kisebb, értelmes egységekre, tokenekre bontjuk, amelyek lehetnek szavak, karakterek vagy kifejezések. Ez az első és alapvető lépése a legtöbb NLP feladatnak. A kód a kínai nyelvre koncentrál, ahol a szavak között nem mindig találunk szóközöket, ami megnehezíti a hagyományos tokenizálási módszerek alkalmazását. A Jieba könyvtár egy kifejezetten a kínai nyelvhez készült eszköz, amely képes megbízhatóan felismerni a kínai szavakat és kifejezéseket, még szóközök hiányában is. A kódban először betöltjük a Jieba könyvtárat, majd megadunk egy kínai nyelvű szöveget. A jieba.cut() függvény segítségével ezt a szöveget tokenekre bontjuk, és a kapott tokeneket egy listában jelenítjük meg. Így a kód bemutatja, hogyan lehet egy olyan nyelvet, amelynek más a szerkezete, mint az európai nyelvek, hatékonyan tokenizálni egy speciális eszköz segítségével.

```python
import jieba
text = "我来到北京清华大学"
tokens = jieba.cut(text)
print(list(tokens))
```

15. ábra Jieba modell

['我', '来到', '北京', '清华大学']

8.2 Korpusok

8.2.1 Korpusok fogalma és jelentősége

A korpusok nagy mennyiségű szöveges adatgyűjtemények, amelyeket az NLP-modellek betanítására használnak. A korpusok minősége és mérete jelentős hatással van a modellek teljesítményére. A korpusok lehetnek általános célúak vagy specifikus feladatokra szántak.

8.2.2 Ismert korpuszok és alkalmazásuk

- **Wikipedia**: Gyakran használt korpusz általános nyelvi modellek betanítására. A Wikipedia adatai gazdagok és változatosak, ami hasznos a nyelvi modellek általános képességeinek fejlesztésére.
- **Common Crawl**: Egy hatalmas webes adatgyűjtemény, amelyet gyakran használnak mélytanulási modellek betanítására. Különösen hasznosak, mivel tartalmazzák a web széles körű tartalmait, beleértve a különböző nyelveket és témákat.
- **IMDB Reviews**: Specifikus korpusz, amelyet gyakran használnak szentimentelemzésre. Az IMDB Reviews adatai tartalmazzák a filmkritikák szövegeit és azokhoz tartozó értékeléseket, ami lehetővé teszi a modellek számára az érzelmi tartalom felismerését és elemzését.
- **Speciális korpuszok**: A szakszövegek és domain-specifikus adatbázisok, fontosak a specifikus feladatokhoz . Például a biológiai szövegek korpuszai hasznosak a biomedikai kutatásokban, míg a jogi dokumentumok korpuszai a jogi elemzésekben.

8.2.3 Korpuszkészítés és előkészítés

A korpuszok előkészítése magában foglalja az adatok tisztítását, normalizálását és annotálását. Ez a folyamat kulcsfontosságú a modell teljesítményének biztosítása érdekében. Az lenti Python kód egy egyszerű korpusz előkészítésének alapvető lépéseit mutatja be. A korpusz lényegében egy nagy mennyiségű szöveges adat gyűjteménye, amelyet gépi tanulási modellek betanítására használunk. Az előkészítés során a nyers szövegeket olyan formába hozzuk, amely alkalmas a további feldolgozásra.

A kód először definiálja a szövegek listáját, amelyek a korpuszt alkotják. Ezután egy függvényt hoz létre a szövegek tisztítására és normalizálására. A tisztítás során a szöveget kisbetűssé alakítja, eltávolítja a nem alfanumerikus karaktereket (pl. írásjelek, számok) és a felesleges szóközöket. A tisztított szövegeket egy új listában tárolja.

```python
import re
# Szövegek listája
texts = ["Hello, world!",
    "Natural Language Processing with Python.",
    "Tokenization is a important step in NLP."]
# Tisztítás és normalizálás
def clean_text(text):
    text = text.lower()  # Kisbetűsre alakítás
    text = re.sub(r'\W', ' ', text)  # Nem alfanumerikus karakterek eltávolítása
    text = re.sub(r'\s+', ' ', text)  # Többszörös szóközök eltávolítása
    return text
cleaned_texts = [clean_text(text) for text in texts]
print(cleaned_texts)
```

16. ábra Korpusz előkészítés

['hello world ', 'natural language processing with python ', 'tokenization is a important step in nlp ']

8.2.4 Korpuszok Létrehozása és Használata

Az adatgyűjtés és annotálás magában foglalja a korpuszok létrehozását. Az adatgyűjtés során különböző forrásokból származó szövegeket gyűjtünk össze, míg az annotálás során a szövegeket címkékkel látjuk el, amelyek segítik a modell betanítását.

A lentebb lévő Python kód egy korpusz létrehozásának első lépcsőjét, az annotálást szemlélteti a **Label Studio** könyvtár segítségével. A Label Studio egy interaktív adatjelölő eszköz, amely lehetővé teszi, hogy a felhasználó közvetlenül részt vegyen a szövegek címkézésében.

A parancs végrehajtása után a Label Studio egy webes felületet nyit meg, ahol a felhasználó képernyőn megjelenő szövegrészeket látva manuálisan címkézheti a benne szereplő entitásokat (például személyneveket, szervezeteket, helyeket). A felhasználó által adott címkéket a rendszer elmenti, és ezek alapján egy annotált adatkészlet jön létre, amelyet később gépi tanulási modellek betanítására lehet használni.

```
# Telepítsd a Label Studio-t
!pip install label-studio

# Indítsd el a Label Studio-t
!label-studio start
```

17. ábra Label studio

9. TRANSZFERTANULÁS ÉS A NAGY NYELVI MODELLEK (LLM)

9.1 A Transzfer Tanulás Alapjai

9.1.1 A transzfer tanulás koncepciója és előnyei

A transzfer tanulás egy olyan technika, amelyben egy előre betanított modell tudását új feladatokra alkalmazzák. Hatékonyan alkalmazható kevés adatmennyiséggel rendelkező feladatokra, gyorsabb és olcsóbb modellek fejlesztését teszi lehetővé. A transzfer tanulás különösen hasznos olyan esetekben, amikor kevés adat áll rendelkezésre a modell betanításához. Például a ritkán beszélt nyelvek esetében a rendelkezésre álló szöveges adatok gyakran korlátozottak, így a transzfer tanulás lehetővé teszi, hogy más nyelveken szerzett tudást felhasználva hatékonyabb modelleket hozzunk létre. A transzfer tanulás egyik előnye, hogy lehetővé teszi a nagy adatbázisokon betanított modellek finomhangolását kisebb, specifikus adatbázisokra, ami jelentősen növeli a modell pontosságát és hatékonyságát

9.1.2 Előre betanított modellek (pl. BERT, GPT)

Számos előre betanított modell áll rendelkezésre, amelyeket különböző NLP-feladatokra lehet használni. Ezek a modellek nagy mennyiségű adatokon lettek betanítva, és hatékony kiindulópontot jelentenek az új modellek fejlesztéséhez. Például a BERT és a GPT modellek különösen hatékonyak a kontextus figyelembevételében és a természetes nyelvi szövegek generálásában. Az előre betanított modellek használata jelentősen csökkenti a szükséges adatmennyiséget és a fejlesztési időt, mivel a modell már rendelkezik egy alapvető nyelvi tudással, amelyet csak finom hangolni kell az új feladathoz

9.1.3 A transzfer tanulás lépései

1. **Előre betanított modell kiválasztása**: A modellnek relevánsnak kell lennie a feladathoz. Például a BERT modellt választhatjuk szövegklasszifikációs feladatra.
2. **Modell betöltése**: Betöltés a számítógépre és a szükséges függőségek telepítése.
3. **Adatok előkészítése**: Az adatok előkészítése a modell számára, ami magában foglalja az adatok tisztítását, normalizálását és formázását.
4. **Finomhangolás (fine-tuning)**: A modell finomhangolása a saját adatokon, ami a modell paramétereinek beállítását jelenti a feladathoz való jobb illeszkedés érdekében.
5. **Értékelés és finomhangolás**: További finomhangolásra is szükség lehet.
6. **Bevezetés**: A modell bevezetése, alkalmazása, monitorozása, és karbantartása.

9.2 Nagy Nyelvi Modellek (LLM)

9.2.1 Az LLM definíciója és jellemzői

A nagy nyelvi modellek (LLM-ek) mély neurális hálózatok, amelyek hatalmas mennyiségű szöveges adatból tanulnak, és képesek emberi szintű nyelvi feladatok elvégzésére. Az LLM-ek a természetes nyelv feldolgozásának (NLP) területén jelentős áttörést jelentenek, mivel képesek összetett nyelvi struktúrákat megérteni és generálni.

Ezek a modellek (LLM-ek) mint a GPT-4, BERT és a Transformer alapú modellek, az utóbbi években jelentős előrelépést hoztak az NLP területén. Mivel hatalmas adathalmazokon lettek betanítva, és képesek komplex nyelvi feladatok megoldására. Az LLM-ek különböző alkalmazási területeken bizonyítják hatékonyságukat, beleértve az automatikus szövegírást, a kérdés-válasz rendszereket és a gépi fordítást.

Méret

Az LLM-ek jelentős számítási kapacitást és erőforrást igényelnek. A modellek betanítása és futtatása költséges, és magas technikai követelményeket támaszt. Több milliárd paraméterük lehet, ami lehetővé teszi számukra, hogy bonyolult nyelvi mintázatokat tanuljanak meg. A Google által kifejlesztett BERT és az OpenAI GPT-3 példák arra, hogyan lehet több milliárd paraméterrel rendelkező modelleket hatékonyan felhasználni, hogy mélyebben megértsék a nyelvet, és komplexebb feladatokat oldjanak meg.

Adat

Az LLM-eket hatalmas mennyiségű, változatos szöveges adaton képezik ki, ami széleskörű nyelvi tudást biztosít számukra. Ez az adat lehet például könyvek, cikkek, weboldalak vagy akár közösségi média bejegyzések.

Feladatok

Az LLM-ek képesek különböző nyelvi feladatokat elvégezni, mint például a szöveggenerálás, a fordítás, a kérdés-felelet, a szövegösszefoglalás, a szövegérzékelés és a szöveg kiegészítése.

Általánosíthatóság

Az LLM-ek képesek általánosítani az új adatokra, és új feladatokra alkalmazni a tanultakat. Ez azt jelenti, hogy egy LLM-et, amelyet például angol nyelvű szövegeken képeztek ki, viszonylag könnyen át lehet alakítani más nyelvekre vagy más típusú szövegekre való alkalmazásra.

9.2.2 LLM képzése

Adatmennyiség és -minőség

Az LLM-ek képzéséhez szükséges adatmennyiség és -minőség szorosan összefügg. A hatalmas mennyiségű, változatos és magas minőségű adat biztosítja, hogy a modellek képesek legyenek megérteni és generálni összetett nyelvi struktúrákat. Az adat előkészítése pedig kulcsfontosságú ahhoz, hogy az adatok a lehető leghatékonyabban felhasználhatók legyenek a modell képzésében.

Miért van szükség hatalmas mennyiségű adatra?

- **Mintázatfelismerés:** Az LLM-ek a szöveges adatokban rejlő mintázatokat tanulják meg. Minél több adat áll rendelkezésre, annál több mintázatot képesek azonosítani, és annál pontosabb lesz a nyelvhasználatuk.
- **Általánosítás:** A nagy mennyiségű adat segít az LLM-eknek abban, hogy általánosítsanak az új, eddig nem látott adatokra. Ezáltal képesek lesznek olyan szövegeket generálni vagy értelmezni, amelyek a képzési adatokban nem szerepeltek.
- **Bonyolult feladatok megoldása:** Az összetettebb nyelvi feladatokhoz, például a fordításhoz, a szövegösszefoglaláshoz vagy a kérdés-felelet játékokhoz, még több adat szükséges.

Az adatminőség jelentősége

- **Torzítások elkerülése:** Ha az adatok torzítottak vagy elfogultak, az LLM is átveszi ezeket a torzításokat. Például, ha a képzési adatokban túlnyomórészt egy bizonyos nézőpont vagy stílus jelenik meg, akkor a modell is inkább ezt fogja preferálni.
- **Reális kép a nyelvről:** A magas minőségű adatok olyan nyelvi jelenségeket tartalmaznak, amelyek a valós életben is előfordulnak. Ez segít az LLM-eknek abban, hogy jobban megértsék a nyelv működését és a természetes nyelvű párbeszédek bonyolultságát.
- **Konzisztens teljesítmény:** A jó minőségű adatok biztosítják, hogy az LLM konzisztens teljesítményt nyújtson különböző feladatokban és különböző szövegeken.

Az adat előkészítése

- **Adattisztítás**: Az adattisztítás során a hibás, hiányos vagy irreleváns adatokat távolítjuk el az adathalmazból. A tisztítás célja, hogy a modell minőségi adatokon tanulhasson, és ne torzuljanak el az eredmények a hibás adatok miatt.

- **Tokenizálás**: A tokenizálás során a szöveget kisebb egységekre, úgynevezett tokenekre bontjuk. Ezek a tokenek lehetnek szavak, számjegyek vagy egyéb karakterek. A tokenizálás megkönnyíti a számítógép számára a szöveg feldolgozását, és lehetővé teszi, hogy a modell szintenként elemezze a szöveget.

- **Normalizálás**: A normalizálás során az adatokat egységes formátumra hozzuk. Ez magában foglalhatja a kisbetűssé alakítást, az elírások javítását, a stop szavak eltávolítását és a szavak származtatását. A normalizálás célja, hogy az adatokat konzisztenssé tegyük, és csökkentsük a szóformák variációit.

- **Feature engineering:** A feature engineering során új jellemzőket hozunk létre az adatokból, amelyek segíthetnek a modellnek jobban megérteni a szöveget. Ezek a jellemzők lehetnek például n-grammok (szavak egymás utáni sorozatai), TF-IDF (a szavak fontosságát mérő érték) vagy word embeddings (a szavak vektoros reprezentációja). A feature engineering segítségével a modell képes lehet felismerni olyan összefüggéseket, amelyek az eredeti adatokban nem voltak ott közvetlenül.

Képzési módszerek:

- **Felügyelet nélküli tanulás:** A modell nagy mennyiségű szöveges adatból önállóan tanul meg nyelvi reprezentációkat. Ez a módszer lehetővé teszi a modell számára, hogy a nyelv szerkezetét és a szavak közötti kapcsolatokat megértse.
- **Felügyelt finomhangolás:** A modellt egy konkrét feladatra (pl. szövegosztályozás) finomhangolják egy kisebb, címkézett adathalmazon. Ez a módszer lehetővé teszi, hogy a modell a feladathoz specifikus tudást szerezzen.

Kihívások:

- **Túltanulás:** Ha a modell túl sok figyelmet szentel a képzési adatoknak, akkor nem lesz képes általánosítani az új adatokra.
- **Torzítások:** Az LLM-ek a képzési adatokban jelenlévő torzításokat is megtanulhatják, ami elfogult eredményekhez vezethet.

9.2.4 LLM alkalmazási területei

A nagy nyelvi modellek (LLM) rendkívül sokrétű alkalmazási lehetőségeket kínálnak a különböző iparágakban. Nézzük meg részletesebben az általad említett területeket, és fejtsük ki, hogy miért olyan hatékonyak az LLM-ek ezen a téren:

Automatikus szövegírás

- **Kreatív írás:** Az LLM-ek képesek versek, forgatókönyvek, teljes regények írására.
- **Tartalomgenerálás:** Hírek, blogok, marketing anyagok automatikus generálása.
- **Personalizált kommunikáció:** E-mailek, üzenetek személyre szabása az egyes felhasználók érdeklődése alapján.

Példa: A GPT-3 képes olyan szövegeket generálni, amelyek alig különböztethetők meg az ember által írt szövegektől.

Kérdés-válasz rendszerek

- **Információkeresés:** A felhasználók kérdéseire pontos és releváns válaszokat adnak.
- **Ügyfélszolgálat:** Automatizált ügyfélszolgálati rendszerekben használhatóak.

Példa: A Google keresője egyre inkább támaszkodik az LLM-ekre, hogy pontosabb és relevánsabb válaszokat adjon a felhasználóknak.

Gépi fordítás

- **Valós idejű fordítás:** Szövegek, beszéd fordítása más nyelvekre.
- **Domain-specifikus fordítás:** Szakmai szövegek (pl. orvosi, jogi) pontos fordítása.
- **Multilingvális kommunikáció**: A különböző nyelveket beszélő emberek közötti kommunikáció megkönnyítése.

Példa: A Google Translate az egyik legismertebb fordító rendszer, amely LLM-eken alapul.

Chatbotok és virtuális asszisztensek

- **Ügyfélszolgálat:** 24/7 elérhető ügyfélszolgálat, gyors válaszok.
- **Személyes asszisztens:** Feladatkezelés, időpontfoglalás, emlékeztetők.
- **Oktatás:** Interaktív tanulási élmény nyújtása.

Példa: Az Apple Siri és az Amazon Alexa olyan virtuális asszisztensek, amelyek LLM-ek segítségével értik meg a felhasználók utasításait.

Szövegosztályozás

- **Érzelemfelismerés:** A szövegben kifejezett érzelmek azonosítása.
- **Témák szerinti besorolás:** A szöveg témájának meghatározása (pl. politika, sport, technológia).
- **Spam szűrés:** A nem kívánt e-mailek kiszűrése.

Példa: A Twitter képes automatikusan azonosítani a gyűlöletbeszédet tartalmazó tweeteket az LLM-ek segítségével.

Az LLM-ek jövője

Az LLM-ek fejlődése rendkívül gyors, és várhatóan még sok új és izgalmas alkalmazási területük lesz a jövőben. A folyamatos fejlesztéseknek köszönhetően az LLM-ek egyre intelligensebbek, hatékonyabbak és sokoldalúbbak lesznek. A jövőben várhatóan olyan területeken is jelentős szerepet fognak játszani, mint az egészségügy, a jog, vagy akár a művészet. Az LLM-eknek köszönhetően személyre szabottabb szolgáltatásokhoz juthatunk, hatékonyabbá válhatnak az oktatási folyamatok, és új lehetőségek nyílnak meg a kutatásban is. Ugyanakkor a gyors fejlődés számos etikai kérdést is felvet, mint például az adatvédelem, a felelősség, vagy a mesterséges intelligencia esetleges visszaélése. Fontos, hogy a fejlesztők és a szabályozók együttműködjenek annak érdekében, hogy az LLM-ek pozitív hatással legyenek a társadalomra.

9.2.5 LLM korlátai és etikai kérdések

Az LLM-ek hatalmas potenciállal rendelkeznek, de számos korláttal és etikai kérdéssel is szembesülünk:

Torzítások és elfogultság

Az LLM-ek a képzésük során használt adatokból tanulnak, így a bennük rejlő torzítások és sztereotípiák könnyen átkerülhetnek a generált szövegekbe. Ez diszkriminatív vagy sértő kimenetekhez vezethet, amelyek súlyos társadalmi következményekkel járhatnak.

A felelősség kérdése

Ha egy LLM hamis információt terjeszt, vagy kárt okoz, ki a felelős? A fejlesztő, a felhasználó, vagy maga a modell? A felelősség kérdésének tisztázása kulcsfontosságú a technológia biztonságos és etikus alkalmazásához.

Mély hamisítványok

Az LLM-ek segítségével könnyen lehet hamis híreket, mély hamisítványokat vagy manipulatív tartalmakat létrehozni, amelyek megtéveszthetik az embereket és alááshatják a társadalmi bizalmat.

Adatvédelem és biztonság

Az LLM-ek képzéséhez és működéséhez hatalmas mennyiségű adat szükséges, amelyek között gyakran találhatók személyes adatok is. Ennek megfelelően felmerül az adatvédelem és a biztonság kérdése, hiszen a visszaélések súlyos következményekkel járhatnak.

Energiafogyasztás

Az LLM-ek képzése és üzemeltetése rendkívül energiaigényes folyamat, ami jelentős környezeti terhelést jelent. A fenntarthatóság szempontjából elengedhetetlen, hogy olyan megoldásokat találjunk, amelyek csökkentik az LLM-ek energiafogyasztását.

Egy 2019-es tanulmány szerint a GPT-3 betanításához körülbelül 1287 megawattosóra (MWh) elektromos energia kellett. Ez nagyjából annyi, mint amennyit egy átlagos amerikai háztartás 120 év alatt fogyaszt el. - https://www.brusselstimes.com/1042696/chatgpt-consumes-25-times-more-energy-than-google

9.3 RAG Modellek és Architektúrák

9.3.1 Miért van szükség a RAG-ra?

A nagy nyelvi modellek (LLM-ek) rendkívüli képességekkel rendelkeznek a szöveg generálásában és a természetes nyelv megértésében, azonban korlátaik is vannak. Egyik fő problémájuk, hogy hajlamosak a hallucinációra, vagyis olyan információkat generálnak, amelyek nem szerepelnek a képzési adataikban, vagy amelyek egyszerűen nem igazak. Ez különösen akkor jelent problémát, ha az LLM-eket olyan feladatokra használjuk, amelyek pontos és naprakész információkat igényelnek.

A **Retrieval-Augmented Generation (RAG)** egy olyan megközelítés, amely az LLM-ek ezen korlátait igyekszik kiküszöbölni. A RAG lényege, hogy az LLM-et kiegészítjük egy külső adatbázishoz való hozzáféréssel, amely releváns információkat tartalmaz a feladathoz. Így az LLM nemcsak a saját belső tudására támaszkodhat, hanem a külső forrásokból származó információkat is felhasználhatja a válaszok generálásához.

9.3.2 RAG működési elve

Információkeresés

A felhasználó által feltett kérdés alapján a rendszer releváns információkat keres egy külső adatbázisban. Ez az adatbázis lehet egy egyszerű szöveges dokumentumgyűjtemény, vagy egy strukturált adatbázis, amely tartalmazza a szükséges információkat. A keresés során általában valamilyen hasonlósági mértéket használnak, hogy meghatározzák, mely dokumentumok a legrelevánsabbak a kérdéshez.

Generálás

A talált releváns dokumentumokat átadják az LLM-nek, amely felhasználja azokat a válasz generálásához. Az LLM a dokumentumokban található információkat kombinálja a saját belső tudásával, hogy egy koherens és informatív választ adjon.

9.3.3 RAG architektúrák

Egy tipikus RAG rendszer a következő fő komponensekből áll:

Retriever

Ez a komponens felelős a releváns információk kereséséért az adatbázisban. A retrieverek lehetnek egyszerű kulcsszó alapú keresők vagy összetettebb vektoros keresők, amelyek a dokumentumok és a kérdés közötti szemantikai hasonlóságot mérik.

Generator

Ez az LLM, amely a retriever által talált információkat felhasználva generálja a választ. A generator általában egy nagy nyelvi modell, például GPT-3 vagy BERT.

Adatbázis

Ez tartalmazza azokat az információkat, amelyeket a modell a válaszok generálásához felhasználhat. Az adatbázis lehet egy egyszerű szöveges dokumentumgyűjtemény, vagy egy strukturált adatbázis, amely tartalmazza a szükséges információkat.

9.3.4 RAG alkalmazási példák

A RAG számos területen alkalmazható:

- **Ismerőközpontú rendszerek:** A RAG-ot felhasználhatjuk olyan rendszerek fejlesztésére, amelyek képesek pontos és naprakész információkat nyújtani a felhasználóknak. Például egy vállalati tudásbázishoz kapcsolhatunk egy RAG rendszert, amely képes a munkavállalók kérdéseire pontos válaszokat adni.
- **Kreativitást igénylő feladatok:** A RAG segítségével kreatívabb és változatosabb szövegeket generálhatunk. Például egy marketinges használhatja a RAG-ot új szlogenek vagy termékleírások generálására.
- **Chatbotok és virtuális asszisztensek:** A RAG-ot felhasználhatjuk olyan chatbotok és virtuális asszisztensek fejlesztésére, amelyek képesek a felhasználókkal természetesebb és informatívabb beszélgetéseket folytatni.

9.3.5 A RAG előnyei

Pontosság: A RAG-nak köszönhetően az LLM-ek képesek pontosabb és hitelesebb válaszokat generálni, mivel a válaszok a külső adatbázisban található információkon alapulnak.

Érthetőség: A RAG segítségével az LLM-ek képesek jobban megérteni a felhasználók szándékát, mivel a külső információkkal kiegészítve képesek a kontextust figyelembe venni.

Rugalmasság: A RAG-ot különböző típusú adatbázisokkal és LLM-ekkel kombinálhatjuk, így a rendszer könnyen adaptálható különböző feladatokhoz és környezetekhez.

9.3.6 A promtolás szerepe a RAG modellekben

A promtolás a RAG modellek hatékony működésének egyik kulcsfontosságú eleme.

A jól megfogalmazott promptok segítségével a modellek képesek pontosabb, relevánsabb és kreatívabb válaszokat generálni, valamint a felhasználói élményt jelentősen javítani.

Promtolás hasznossága a RAG modellekben:

- **Pontosabb eredmények:** Egy jól megfogalmazott prompt segít a modellnek, hogy a releváns információkat találja meg az adatbázisban, és így pontosabb és relevánsabb válaszokat generáljon.
- **Kreatívabb válaszok:** A promtolás segítségével különböző típusú válaszokat lehet generálni, például kreatív szövegeket, kódot vagy akár művészeti alkotásokat.

- **Csökkentett hallucináció:** A jól megfogalmazott prompt csökkentheti a hallucinációk kockázatát, vagyis az olyan helyzetek számát, amikor a modell olyan információkat generál, amelyek nem szerepelnek a képzési adataiban.
- **Felhasználói élmény javítása:** Egy jól megtervezett promotlási stratégia javíthatja a felhasználói élményt, mivel a felhasználók könnyebben és hatékonyabban kaphatnak választ kérdéseikre.

Hatékony promtolás felhasználás a RAG modellekben:

- **Konkrét és egyértelmű:** A promptnak világosnak és egyértelműnek kell lennie, hogy a modell pontosan megértse, mit vár tőle a felhasználó.
- **Kontextus:** A promptban érdemes megadni a szükséges kontextusinformációkat, például a téma, a cél vagy a kívánt válasz formátuma.
- **Példák:** Ha lehetséges, érdemes példákat is adni a kívánt válasz típusára.
- **Utmutatások:** A promptban megadhatunk olyan utasításokat is, amelyek befolyásolják a válasz stílusát, hangnemét vagy hosszát.
- **Szakmai terminológia:** Ha a feladat szakmai jellegű, érdemes a megfelelő szakkifejezéseket használni a promptban.
- **Negatív példák:** Megadhatunk olyan példákat is, amelyeket a modellnek el kell kerülnie.

Példa:

- **Rossz prompt:** "Mondj el valamit a mesterséges intelligenciáról."
- **Jó prompt:** "Írj egy 300 szavas esszét a nagy nyelvi modellek fejlődéséről az elmúlt 5 évben, különös tekintettel a GPT-3 modellre és annak alkalmazási lehetőségeire a természetes nyelv feldolgozásában."

A második prompt sokkal pontosabb és részletesebb, így a modell sokkal jobb választ tud generálni.

A promtolás folyamata iteratív: A legjobb eredményeket akkor érhetjük el, ha folyamatosan finomítjuk a promptokat, és figyelemmel kísérjük a modell által generált válaszokat.

A promtolás egy komplex és folyamatosan fejlődő terület. A jól megtervezett promptok kulcsfontosságúak ahhoz, hogy a RAG modellek teljes potenciáljukat ki tudják bontakoztatni.

9.4. A Figyelemmechanizmus Szerepe a Nagy Nyelvi Modellekben

A természetes nyelvfeldolgozás (NLP) területén a figyelemmechanizmus (attention mechanism) bevezetése kiemelkedő jelentőséggel bír, és alapvető szerepet játszott a nagy nyelvi modellek (NLP) fejlesztésében. A figyelemmechanizmus az a technika, amely lehetővé teszi a modellek számára, hogy a bemeneti szöveg különböző részeire különböző mértékben figyeljenek, így javítva a nyelvi kontextus megértését. Ez a megközelítés jelentős áttörést jelentett a korábbi architektúrákkal, például a rekurzív neurális hálózatokkal (RNN) és a konvolúciós neurális hálózatokkal (CNN) szemben.

9.4.1 A Figyelemmechanizmus Működése

A figyelemmechanizmus alapvetően a bemeneti szekvenciák elemeinek súlyozását jelenti, lehetővé téve a modell számára, hogy a legfontosabb információkra fókuszáljon. A figyelemmechanizmus általában a következő lépésekből áll:

1. **Bemeneti Reprezentációk Képzése**: A bemeneti szekvenciát, például egy mondatot, vektorok formájában reprezentálják, ahol minden vektor a mondat egy-egy szavát reprezentálja.
2. **Súlyok Számítása**: A figyelemmechanizmus kiszámítja a súlyokat, amelyek megmutatják, hogy a bemeneti vektorok mely részei a legfontosabbak az adott kontextusban. Ezt a súlyozást a "softmax" függvény segítségével végzik, amely biztosítja, hogy a súlyok összege 1 legyen.
3. **Súlyozott Összegzés**: A bemeneti vektorok súlyozott összegzése történik, ahol a fontosabb szavak nagyobb súllyal szerepelnek, így a modell a releváns információkat emeli ki.
4. **Kimeneti Reprezentációk Generálása**: Az így kapott súlyozott vektorokat felhasználva a modell képes a következő lépéseket végrehajtani, például a következő szó előrejelzésére vagy a szöveg fordítására.

9.4.2 A Figyelemmechanizmus Típusai

A figyelemmechanizmus különböző típusai léteznek, amelyek eltérő célokat szolgálnak. Két alapvető típusa a "self-attention" és a "cross-attention".

- **Self-Attention**: A self-attention mechanizmus során a bemeneti szekvencia minden eleme figyelmet kap a szekvencia többi elemétől. Ez a megközelítés lehetővé teszi a kontextuális kapcsolatok kiemelését és a hosszú távú függőségek kezelését. Például a BERT modell a self-attention mechanizmust alkalmazza, hogy képes legyen a mondatok belső összefüggéseit figyelembe venni. (Ashish Vaswani, 2023)
- **Cross-Attention**: A cross-attention mechanizmus akkor alkalmazható, amikor a modellnek különböző forrásokból kell információt integrálnia. Ez jellemző a dekóder architektúrákra, például a neurális gépi fordítókban, ahol a bemeneti szekvencia és a kimeneti szekvencia között interakció van.

9.4.3 Figyelemmechanizmus Példákon Keresztül

Vegyük példaként egy szövegfordítási feladatot, ahol a figyelemmechanizmus lehetővé teszi, hogy a modell helyesen azonosítsa, mely szavak kapcsolódnak egymáshoz különböző nyelvek között. Például a „The cat is on the mat” mondat fordítása során a modell a „cat” és a „mat” szavak közötti kapcsolatot is felismeri, így biztosítva a pontos fordítást. Ez a szelf-attention képesség lehetővé teszi, hogy a modell minden szóra figyelmet fordítson, nemcsak a közvetlenül előtte vagy utána lévő szavakra.

9.4.4. Figyelemmechanizmus és Nagy Nyelvi Modellek

A figyelemmechanizmus megjelenése óta a nagy nyelvi modellek teljesítménye drámaian javult. A transzformerek architektúrája, amely a figyelemmechanizmusra épít, lehetővé teszi a párhuzamos feldolgozást, így jelentős mértékben felgyorsítja a tanulási folyamatot. Az ilyen modellek, mint például a BERT és a GPT-3, képesek a nyelvi kontextus mélyebb megértésére, és ezáltal jobb eredményeket produkálnak különböző NLP feladatok során, mint például a szövegklasszifikáció, a gépi fordítás és a kérdés-válasz rendszerek.

A figyelemmechanizmus alkalmazásának köszönhetően a modellek képesek az összes bemeneti elemre figyelni, függetlenül azok pozíciójától a szekvenciában, így a hosszú távú függőségek kezelésére is képesek. Ez a képesség elengedhetetlen a természetes nyelv megértésében, amely gyakran igényli, hogy a szavak és kifejezések jelentése a szöveg teljes kontextusában legyen értelmezve.

9.5. A Transzformerek szerepe a nagy nyelvi modellekben

A természetes nyelvfeldolgozás (NLP) területén az utóbbi évek egyik legfontosabb fejlődése a transzformerek architektúrájának bevezetése volt. A transzformerek a "Attention is All You Need" című tanulmányban jelentek meg, és forradalmasították a nyelvi modellek fejlesztését.

Mielőtt a transzformerekről bővebben beszélnénk, érdemes áttekinteni a nyelvi modellek hagyományos megközelítéseit és azok korlátait. (Ashish Vaswani, 2023)

A hagyományos nyelvi modellek, mint például a n-gram modellek és a rekurzív neurális hálózatok (RNN), gyakran szenvednek a hosszú távú függőségek kezelésének nehézségeitől. Az RNN-ek, bár képesek a szekvenciák feldolgozására, hajlamosak voltak az információ "eltűnésére" a hosszabb szövegekben. Ezen problémák kiküszöbölésére érkeztek a transzformerek, képesek a párhuzamos feldolgozásra és a globális figyelem fenntartására.

9.5.1. A Transzformerek Architektúrája

A transzformerek alapja az "attention" mechanizmus, amely lehetővé teszi a bemeneti szekvenciák különböző részeire való figyelem összpontosítását. Ez a mechanizmus két fő komponensből áll: a "self-attention" és a "cross-attention" mechanizmusból. A self-attention lehetővé teszi, hogy a modell figyeljen a bemeneti szekvencia minden egyes elemére, amikor egy adott elem reprezentációját generálja. A cross-attention viszont a dekóder részben működik, lehetővé téve a bemeneti és a kimeneti szekvenciák közötti interakciót.

A transzformerek architektúrája lehetővé teszi, hogy a modellek párhuzamosan dolgozzanak fel az adatokat, így jelentősen felgyorsítva a tanulási folyamatot. A hagyományos RNN-ekkel ellentétben, amelyek lineárisan, lépésről lépésre dolgozzák fel a bemenetet, a transzformerek képesek az összes bemeneti elem egyidejű feldolgozására, ami jelentősen javítja a hatékonyságot (Dai et al., 2019).

9.5.2. A Hugging Face és a Transzformerek

A Hugging Face egy olyan platform, amely a transzformerek alkalmazására és fejlesztésére összpontosít. Az általuk kifejlesztett "Transformers" könyvtár lehetővé teszi a kutatók és fejlesztők számára, hogy könnyen használhassák a legújabb transzformer alapú modelleket. A Hugging Face számos előképzett modellt kínál, mint például a BERT, GPT-2, RoBERTa és T5, amelyek különféle nyelvi feladatokhoz alkalmazhatóak.

A BERT (Bidirectional Encoder Representations from Transformers) egy forradalmi modell, amely a kétirányú figyelmi mechanizmus révén képes a kontextuális jelentések megértésére.

A BERT előképzése két fő feladatra épül: a maszkos nyelvi modellezés és a következő mondat előrejelzése. Ez lehetővé teszi a modell számára, hogy mélyebb és gazdagabb nyelvi reprezentációkat építsen fel, amelyek jól alkalmazhatóak például a szövegklasszifikációs feladatokhoz (Devlin et al., 2019).

A GPT (Generative Pre-trained Transformer) modell, más néven a generatív előképzett transzformer, a szöveg generálására specializálódik. A GPT-2 és a GPT-3 modellek a nagyméretű adathalmazon végzett előképzésük révén képesek a szöveg folytonos generálására, valamint a kontextus figyelembevételére. Ezek a modellek különösen hasznosak a kreatív írásban, chatbotokban és más, szövegalapú alkalmazásokban (Radford et al., 2019).

9.5.3. Alkalmazások és Jövőbeli Kilátások

A transzformerek és a Hugging Face által kifejlesztett modellek széleskörű alkalmazásokat találnak a természetes nyelvfeldolgozás területén. A szövegklasszifikáció, a kérdés-válasz rendszerek, a szöveggenerálás, a gépi fordítás és a szentiment analízis mind olyan területek, ahol a transzformerek kiemelkedő teljesítményt nyújtanak.

A Hugging Face platformja lehetővé teszi a kutatók és fejlesztők számára, hogy megosszák munkáikat, tanuljanak egymástól, és hozzájáruljanak a transzformerek fejlődéséhez. A Hugging Face Model Hub egy hatalmas könyvtár, amely számos előképzett modellt és egyéb erőforrást tartalmaz, így a felhasználók könnyen hozzáférhetnek a legújabb fejlesztésekhez.

A jövőbeli kilátások ígéretesek, mivel a transzformerek architektúrája folyamatosan fejlődik, és új modellek, mint például a T5 (Text-to-Text Transfer Transformer) és a BART (Bidirectional and Auto-Regressive Transformers) megjelenésével még szélesebb lehetőségek nyílnak meg a kutatók előtt. A T5 modell, amely a szöveget egyetlen, egységes feladatra (szöveg-szöveg transzformáció) tanítja, rendkívül sokoldalú és sokféle nyelvi feladatot képes ellátni (Raffel et al., 2020).

9.5.5 A Transzformerek és a Figyelemmechanizmus Hatása az NLP Fejlődésére

A transzformerek és a figyelemmechanizmus forradalmasították a természetes nyelvfeldolgozást. A hagyományos modellekkel összehasonlítva a transzformerek sokkal hatékonyabbak, és mélyebb nyelvi összefüggéseket képesek felfedezni. A figyelemmechanizmus segítségével a modellek képesek nagy mennyiségű szöveget egyszerre feldolgozni, és kontextuálisan releváns információkat nyerni ki belőlük. Ez a rugalmasság

vezetett el olyan áttörésekhez, mint a BERT vagy a GPT modellek, amelyek ma már alapvető eszközei az NLP-nek.

Ez az innováció lehetővé tette, hogy az LLM-ek, például a GPT-3 és GPT-4, különösen jól teljesítsenek szöveggenerálás, kérdés-válasz, és kreatív írás feladatokban, és meghatározóak legyenek a modern mesterséges intelligencia kutatásában és alkalmazásában.

9.4 LLM-ek Hatásai és Kihívásai

9.4.1 Energiafogyasztás és környezeti hatás

Az LLM-ek betanítása hatalmas energiafelhasználással jár, ami jelentős környezeti hatással bír. Az energiafelhasználás optimalizálása és a zöld technológiák alkalmazása fontos a fenntartható fejlesztés érdekében. A modell tréningelésének és használatának energiahatékonyságának növelése érdekében különböző optimalizálási technikákat alkalmaznak, mint például a méretének csökkentése és a hatékonyabb hardverek használata.

9.4.2 Modellek közötti integráció

A különböző LLM-ek integrálása egy komplex rendszerbe kihívást jelent. A modellek közötti interoperabilitás és az adatok konzisztens feldolgozása kritikus tényezők a sikeres alkalmazások szempontjából. Az interoperabilitás biztosítása érdekében standardizált adatcserélő formátumokat és interfészeket kell használni.

9.4.3 LLM-ek kifejlesztésének költségei és partnerei

A GPT-3 kifejlesztése során az OpenAI jelentős pénzügyi és technológiai támogatást kapott. A modell kifejlesztése több millió dollárba került, és a processzoridő energiaigénye hatalmas volt. A Microsoft, mint stratégiai partner, nemcsak pénzügyi támogatást nyújtott, hanem az Azure infrastruktúrát is biztosította a modell betanításához és futtatásához.

9.4.4 Használati kockázatok

Az LLM-ek használata számos kockázattal jár, mint például az adatvédelem, az etikai kérdések és a modellek elfogultsága. Fontos, hogy a fejlesztők és felhasználók figyelembe vegyék ezeket a kockázatokat és megfelelően kezeljék őket.

10. FEJLETT NLP TECHNIKÁK

10.1. Előre betanított modellek

Az előre betanított modellek, mint a BERT és a GPT-3, nagy mennyiségű adatokon lettek betanítva, és hatékonyan alkalmazhatók különböző NLP-feladatokra. Ezek a modellek már rendelkeznek egy alapvető nyelvi tudással, amelyet specifikus feladatokra lehet finomhangolni.

10.2 Finomhangolás specifikus feladatokra

Az előre betanított BERT modell már rendelkezik egy általános nyelvi megértéssel, de a finomhangolás során a modell a specifikus feladatunkra vonatkozó adatokon tanul, így képes lesz a bemeneti szövegeket a kívánt kategóriákba sorolni (például pozitív vagy negatív). Ez a folyamat lehetővé teszi, hogy a BERT modellt számos különböző szöveges feladatra adaptáljuk, például érzelem-elemzésre, szövegösszegzésre vagy kérdés-felelet rendszerekre.

A lenti Python kód egy BERT modell finomhangolását mutatja be egy szövegklasszifikációs feladatra**.** Ez azt jelenti, hogy egy általános, előre betanított nyelvi modellt (BERT) úgy alakítunk át, hogy egy konkrét feladatra, például szövegek pozitív vagy negatív érzelmi töltésének meghatározására legyen képes.

A kód lépései

1. Adatok előkészítése: A szövegeket és a hozzájuk tartozó címkéket (pl. pozitív/negatív) betöltjük és tokenizáljuk, vagyis szavakra bontjuk és numerikus reprezentációba alakítjuk, hogy a modell meg tudja dolgozni. Fontos megjegyezni, hogy az adatok előkészítése magában foglalhatja az adatok tisztítását is, például a fölösleges karakterek eltávolítását.

2. Modell betöltése: Egy előre betanított BERT modellt töltünk be, amely már képes alapvető nyelvi feladatok elvégzésére. A BERT modell architektúrája különösen jól működik a szövegkörnyezet megértésében, mivel a bidirekcionális figyelem mechanizmust alkalmaz.

3. Finomhangolási beállítások**:** Meghatározzuk a finomhangolás paramétereit, például az epochok számát (hányszor megyünk végig az összes adaton), a batch méretét (hány adatpéldát dolgozunk fel egyszerre), és a mentés helyét. Ezen kívül érdemes figyelembe venni a tanulási rátát is, amely befolyásolja a modell tanulásának ütemét.

4. Trainer inicializálása: Létrehozunk egy Trainer objektumot, amely a finomhangolási folyamatot kezeli. A Trainer osztály számos beépített funkcióval rendelkezik, mint például a modell mentése, a metrikák nyomon követése és a validációs eredmények megjelenítése.

5. Modell betanítása**:** A trainer.train() metódus segítségével elindítjuk a finomhangolási folyamatot. A modell a megadott adatokon tanul, és a paramétereit úgy módosítja, hogy a lehető legjobban elvégezze a szövegklasszifikációs feladatot. A tanulási folyamat során a modell folyamatosan finomítja a súlyait, hogy minimalizálja a hibát a predikciók során.

```python
import torch
from torch.utils.data import Dataset, DataLoader
from transformers import BertTokenizer, BertForSequenceClassification, Trainer, TrainingArguments

# Definiálj egy Dataset osztályt
class StoryDataset(Dataset):
    def __init__(self, texts, labels, tokenizer, max_length=512):
        self.texts = texts
        self.labels = labels
        self.tokenizer = tokenizer
        self.max_length = max_length

    def __len__(self):
        return len(self.texts)

    def __getitem__(self, idx):
        text = self.texts[idx]
        label = self.labels[idx]
        encoding = self.tokenizer(
            text,
            truncation=True,
            padding='max_length',
            max_length=self.max_length,
            return_tensors='pt'
        )
        return {
            'input_ids': encoding['input_ids'].flatten(),
            'attention_mask': encoding['attention_mask'].flatten(),
            'labels': torch.tensor(label, dtype=torch.long)
        }
```

18. ábra Finomhangolás Bert

```python
# Adatok betöltése és előkészítése
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
texts = ["Your collection of stories..."]
labels = [0]  # Például, ha több kategóriát is szeretnénk használni

# Hozz létre egy Dataset példányt
dataset = StoryDataset(texts, labels, tokenizer)

# Modell betöltése
model = BertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=2)

# Finomhangolás beállításai
training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=3,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    warmup_steps=500,
    weight_decay=0.01,
    logging_dir='./logs',
)

# Trainer inicializálása
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=dataset,
    eval_dataset=dataset  # Külön validációs dataset ha lheetséges
)

# Modell betanítása
trainer.train()
```

19. ábra Finomhangolás Bert folyt.

10.3 Etikus és tisztességes modellezés

10.3.1 Etikai kérdések és kihívások

Az NLP modellek fejlesztése és alkalmazása során különös figyelmet kell fordítani az etikai kérdésekre, mint például a nemi, faji és kulturális elfogultságok minimalizálására. Az NLP modellek etikus alkalmazása elengedhetetlen a társadalmi felelősségvállalás és a modellek széleskörű elfogadása szempontjából.

10.3.2 Az elfogultságok felismerése és kezelése

Az elfogultságok azonosítása és kezelése a gépi tanulási modellek fejlesztésének egyik legfontosabb szempontja. Ezek a modellek ugyanis a betanításuk során használt adatokból tanulnak, és így azokban jelen lévő torzításokat is átvehetik. Az elfogultságok jelenléte a modellekben számos problémához vezethet, például diszkriminációhoz, igazságtalan döntésekhez és a modell megbízhatóságának csökkenéséhez.

A lentebb megtalálható Python kód egy egyszerű példát mutat be az elfogultság detektálására egy előre betanított modellen. A fill-mask pipeline segítségével egy olyan modellt hozunk

létre, amely képes kitölteni a hiányzó szavakat egy mondatban. Ha a modellünkben elfogultság van, akkor a kitöltött szavakban is megnyilvánulhat ez. Például, ha a modellünk nemi sztereotípiákat tanult, akkor a "A doktor ..." kiegészítéskor nagyobb valószínűséggel javasol férfias foglalkozásokat.

```
A javasolt kódra licenc vonatkozhat |
from transformers import pipeline

# Elfogultság detektálása egy előre betanított modellen
unmasker = pipeline('fill-mask', model='bert-base-uncased')
result = unmasker("The doctor was a [MASK].")
print(result)
```

20. ábra Unmasker

[{'score': 0.08274320513010025, 'token': 11067, 'token_str': 'genius', 'sequence': 'the doctor was a genius.'}, {'score': 0.07349821925163269, 'token': 7966, 'token_str': 'fool', 'sequence': 'the doctor was a fool.'}, {'score': 0.024674762040376663, 'token': 3460, 'token_str': 'doctor', 'sequence': 'the doctor was a doctor.'}, {'score': 0.023262616246938705, 'token': 4393, 'token_str': 'vampire', 'sequence': 'the doctor was a vampire.'}, {'score': 0.022237030789256096, 'token': 2658, 'token_str': 'professional', 'sequence': 'the doctor was a professional.'}]

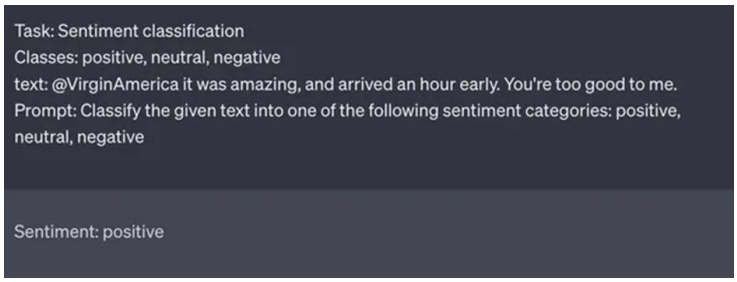
10.4 Zero-shot és few-shot tanulás

A zero-shot és few-shot tanulás olyan gépi tanulási technikák, amelyek lehetővé teszik a modellek számára, hogy új feladatokat minimális vagy akár nulla mennyiségű, kifejezetten erre a feladatra címkézett adattal tanuljanak meg. Ez rendkívül hasznos olyan esetekben, amikor kevés tanító adat áll rendelkezésre, vagy amikor új kategóriák jelennek meg, amelyekre nincsenek előre betanított modellek. A zero-shot tanulásban a modell képes új kategóriákat felismerni anélkül, hogy bármilyen példát látott volna róla a képzés során, míg a few-shot tanulásban néhány példát kap a modell az új kategóriáról. Az egyik leggyakoribb alkalmazási területük a természetes nyelvfeldolgozás, ahol például a sentiment analízisben használják őket, amikor egy adott szöveg érzelmi töltését szeretnénk meghatározni. A lenti Python kód egy példát mutat be a zero-shot osztályozásra a Hugging Face könyvtár segítségével. Itt a pipeline segítségével létrehozunk egy zero-shot osztályozót, majd egy adott szöveget adunk meg, valamint egy listát a lehetséges címkékről (pozitív, negatív). A modell ezután megpróbálja meghatározni, hogy a szöveg melyik címkéhez tartozik a legjobban, és az eredményt egy szótár formájában adja vissza.: „{'sequence': 'This is a great product.', 'labels': ['positive', 'negative'], 'scores': [0.9981971383094788, 0.0018028444610536098]} ”

10.4.1 Zero shot

„Zero-shot esetén nem adunk meg címkézett adatokat a modellhez, és elvárjuk, hogy a modell egy teljesen új megoldást adjon. Használhatjuk például a ChatGPT-t új feladatok megoldására zéró promttal. Az LLM-ek alkalmazkodni tudnak a még nem látott, mert sok forrás alapján megértik a kontextust”.

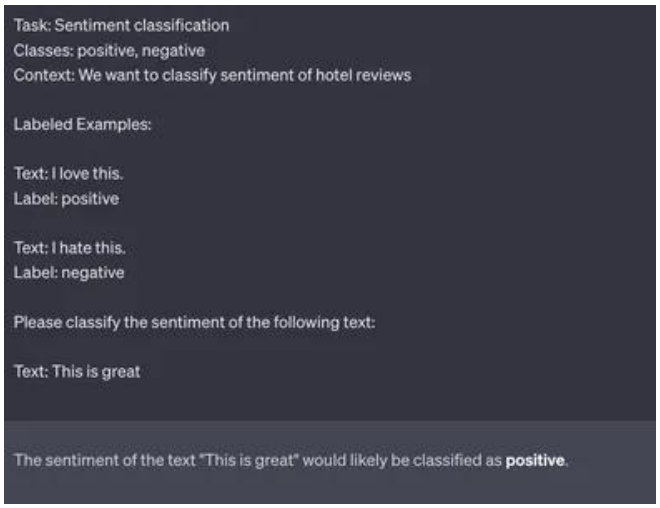
Példa lekérdezés a szöveg pozitív, semleges és negatív érzelmek osztályba sorolására:



21. ábra Zero shot

10.4.2 Few-shot

„A few-shot prompt tartalmaz címkézett példákat. Ez nem ugyanaz, mint a few-shot tanulás, amelyet arra használnak, hogy egy LLM-et példákkal finomhangoljanak egy új probléma esetén. Ez a megközelítés csökkenti a nagy címkézett adathalmazoktól való függést, gyors adaptációt és pontos előrejelzést tesz lehetővé kevés címkézett példával. Akkor célszerű alkalmazni, ha nehéz beszerezni az egyes osztályokba sorolható jelentős méretű mintát.„



22. ábra Few shot

11. TOVÁBBI TECHNIKAI MÓDSZEREK AZ NLP-BEN

11.1 Neurális gépi fordítás (NMT)

11.1.2 A hagyományos statisztikai módszerektől a modern neurális hálózatokig

A gépi fordítás területén a hagyományos statisztikai módszerek (SMT) hosszú ideig uralkodtak. Ezek a módszerek statisztikai modelleket használnak a szavak és kifejezések közötti valószínűségi eloszlások alapján történő fordításra. Az SMT azonban korlátozott volt a fordítási pontosság és a kontextus figyelembevételének szempontjából.

11.1.3 Google Neural Machine Translation (GNMT)

A modern neurális gépi fordítás (NMT) jelentős előrelépést hozott a fordítás minőségében. A GNMT rendszer, amelyet a Google fejlesztett, mélytanulási modelleket használ a szövegek fordítására. Az NMT modellek képesek figyelembe venni a szöveg teljes kontextusát, ami pontosabb és természetesebb fordításokat eredményez. A lenti Python kód egy egyszerű példát mutat be a GNMT használatára. A Transformers könyvtár segítségével betöltünk egy előre betanított modellt, például az angol-német fordításra optimalizált Helsinki-NLP/opus-mt-en-de modellt. A bemeneti szöveget tokenekre bontjuk, és a modell segítségével generálunk egy fordítást. A kapott tokeneket visszaalakítjuk szöveggé, így megkapjuk a fordított mondatot. A GNMT számos területen alkalmazható, például weboldalak, alkalmazások fordításában, chatbotokban, beszédfelismerésben és szöveggenerálásban.

```
from transformers import MarianMTModel, MarianTokenizer
model_name = 'Helsinki-NLP/opus-mt-en-de'
tokenizer = MarianTokenizer.from_pretrained(model_name)
model = MarianMTModel.from_pretrained(model_name)
text = "Hello, how are you?"
translated = model.generate(**tokenizer.prepare_seq2seq_batch([text], return_tensors="pt"))
print([tokenizer.decode(t, skip_special_tokens=True) for t in translated])
```

23. ábra Fordító modell

["Hallo, wie geht's?"]

11.2 Összegzés (Summarization)

Az összegzés, mint a természetes nyelv feldolgozásának egyik központi feladata, két fő típusra osztható: kivonatos és absztrakt összegzésre. A kivonatos összegzésnél a rendszer a bemeneti szöveg legfontosabb mondatait vagy mondatrészleteit választja ki és egyesíti, így egy rövidebb, de a lényeget tartalmazó összefoglalást hoz létre. Ez a módszer különösen hatékony, ha a cél egy szöveg főbb pontjainak gyors áttekintése. Az absztrakt összegzés ezzel szemben új mondatokat generál a szöveg tartalmának összefoglalására, így egy olyan sűrítmény jön létre, amely nem feltétlenül az eredeti szöveg mondatainak egyszerű összekapcsolása. Az absztrakt összegzéshez fejlett mélytanulási modellekre van szükség, amelyek képesek a szöveg jelentésének mélyebb megértésére és új, koherens mondatok generálására.

Mind a kivonatos, mind az absztrakt összegzés számos területen talál alkalmazást, például az újságírásban, ahol a hírek gyors és tömör összefoglalására van szükség, vagy az ügyfélszolgálatban, ahol a felhasználói kérdésekre adott hosszú válaszokat kell röviden és érthetően összefoglalni. A lentebb lévő Python kód egy egyszerű példát mutat be az absztrakt összegzésre a Transformers könyvtár segítségével. Ebben a példában a rendszer egy adott szöveget kap bemenetként, és egy megadott hosszúságú összefoglalást generál róla. Az összegzés során a modell figyelembe veszi a szöveg legfontosabb információit, és egy új, koherens mondatot hoz létre, amely a szöveg lényegét tükrözi. Az összegzés technológiája folyamatosan fejlődik, és várhatóan egyre szélesebb körben elterjed majd a jövőben, mivel számos területen jelentős hatékonyságnövelést eredményezhet.

```
from transformers import pipeline
summarizer = pipeline("summarization")
text = """
In another moment down went Alice after it,
never once considering how in the world she was to get out again.
The rabbit-hole went straight on like a tunnel for some way, and then dipped suddenly down,
so suddenly that Alice had not a moment to think about stopping herself
before she found herself falling down what seemed to be a very deep well.
"""
summary = summarizer(text, max_length=50, min_length=25, do_sample=False)
print(summary)
```

24. ábra Összegzés – Summeraziton

[{'summary_text': ' Alice had not a moment to think about stopping herself before she found herself falling down what seemed to be a very deep well . The rabbit-hole went straight on like a tunnel for some way, and then dipped suddenly down, so suddenly'}]

11.3 Név entitás felismerés (NER)

A név entitás felismerés (NER) egy kulcsfontosságú természetes nyelvfeldolgozási (NLP) feladat, amelynek célja azonosítani és kategorizálni a szövegben szereplő nevesített entitásokat. Ezek lehetnek személyek, szervezetek, helyek, dátumok, pénzösszegek, vagy más, a valóságban létező fogalmak. Az NER-rendszerek képesek például egy mondatban megtalálni és osztályozni a "Budapest", "Apple" vagy "2023. január 1." kifejezéseket.

Az NER-nek számos gyakorlati alkalmazása van. A biológiai adatok elemzésében például segít azonosítani géneket, fehérjéket és betegségeket a tudományos publikációkban. A jogi dokumentumok feldolgozásában a szerződések, bírósági ítéletek vagy törvények elemzésével támogatja a jogi kutatást és az automatizált szerződéskötést. A közösségi média elemzésében pedig a felhasználók véleményének, érzéseinek és a beszélgetések témáinak meghatározásában játszik fontos szerepet.

Az NER-rendszerek általában gépi tanulási modelleken alapulnak, amelyeket nagy mennyiségű címkézett adattal képeznek be. Ezek a modellek képesek megtanulni, hogy milyen szavak vagy kifejezések utalnak egy adott entitásra, és milyen kontextusban jelennek meg ezek az entitások.

A lentebb lévő Python kód egy egyszerű példát mutat be az NER használatára a Hugging Face könyvtár segítségével. A kódban először betöltünk egy előre betanított NER modellt (a dbmdz/bert-large-cased-finetuned-conll03-english modellt), majd egy bemeneti szöveget adunk meg. A modell ezt követően azonosítja a szövegben szereplő entitásokat és azok típusát (például személy, szervezet), és az eredményt egy listában adja vissza.

```
from transformers import pipeline
ner = pipeline("ner", model="dbmdz/bert-large-cased-finetuned-conll03-english")
text = "Alice is a hairdresser at Apple."
entities = ner(text)
print(entities)
```

25. ábra NER

[{'entity': 'I-PER', 'score': 0.99424404, 'index': 1, 'word': 'Alice', 'start': 0, 'end': 5}, {'entity': 'I-ORG', 'score': 0.99784136, 'index': 8, 'word': 'Apple', 'start': 26, 'end': 31}]

12. ESETTANULMÁNY: Gyermek Mese Generátor Fejlesztése Transzfer Tanulással

12.1 Bevezetés

Az AI-alapú történetgenerálás új távlatokat nyit a kreatív tartalomkészítés területén, különösen gyermekek számára szánt mesék létrehozásában. A projekt célja egy olyan modell fejlesztése volt, amely képes gyermekbarát történetek generálására megadott karakterek, helyszínek és események alapján. Az alábbi esettanulmány bemutatja a különböző megközelítéseket, az alkalmazott adatkészletek és modellek értékelését, a kezdeti hibákat, valamint azokat a megoldásokat, amelyek végül sikeres eredményre vezettek.

A dolgozat részletesen elemzi a mese generátor modell fejlesztési folyamatát és annak teljesítményét, a konkrét kódrészletekkel alátámasztva. A vizsgálatot úgy struktúráltam, hogy a modellezéshez használt adatgyűjtés, adatfeldolgozás és finomhangolás minden lényeges aspektusát bemutassa. A cél az, hogy a fejezetek részletesen bemutassák az adatkészlet eredetét, annak megfelelősségét, az elemzési eljárásokat, és a fejlesztett modell különböző verzióit, konkrét kódrészletekkel szemléltetve.

A következő fejezetben bemutatom a rossz modell fejlesztési kísérletét, annak hibáit, valamint azokat a konkrét problémákat, amelyek végül a modell elvetéséhez vezettek. Ezt a fejezetet kiegészítem az elemzések során megfigyelt példákkal és ábrákkal, amelyek bemutatják, hogy miért nem volt megfelelő az adott adatstruktúra a gyermekbarát történetgenerálásra.

12.2 Az alap modell

A GPT-2 modell egy generatív, nagyméretű nyelvi modell, amelyet a Transformer architektúrára épülő, nyílt forráskódú eszközként fejlesztett ki az OpenAI. Alapját a Transformer architektúra önfigyelési (self-attention) mechanizmusa adja, amely kiválóan alkalmas összetett nyelvi minták megértésére és generálására. A GPT-2 modell választása a mese generálás szempontjából több kulcsfontosságú előnnyel járt.

12.2.1.A GPT-2 Alkalmazhatósága Generatív Feladatokra

A GPT-2 modell különösen alkalmas generatív feladatokra, mivel:

- Magas szintű kontextus-megértés: A GPT-2 képes megérteni és követni a szövegek hosszabb távú kontextusát, amely fontos a történetek egységes folytonossága szempontjából. Például a gyermekmesékben gyakran előfordul, hogy a főszereplő egy adott problémával szembesül, amit végül megold. Ezen folyamat bemutatása több bekezdésen átível, így egy olyan modell szükséges, amely képes a történetet egységben tartani.

- Szerkezeti rugalmasság: A GPT-2 modell különböző promptokat és bevezetőket tud kezelni, amelyek egyedi történetstruktúrákat biztosítanak. Ez különösen előnyös, mivel a mese generálás során a történet kezdete, közepe és vége jól körülhatárolt szerkezetet igényel.

12.3.Alternatív Modellek: BERT, T5 és GPT-3

12.3.1 BERT (Bidirectional Encoder Representations from Transformers)

A BERT modell egy kétirányú **encoder-alapú** nyelvi modell, amely a Transformer architektúrán nyugszik. Elsősorban szövegértési és szövegkiegészítési feladatokra optimalizált, mivel az encoder rétegek segítségével a szöveg minden részét teljes kontextusában elemzi. A BERT modell így kiválóan alkalmas olyan feladatokra, mint a kérdés-válasz rendszerek vagy a mondat-klasszifikáció, azonban mivel nem tartalmaz dekóder komponenst, nem képes folytonos szövegalkotásra. Emiatt a BERT nem ideális például történetgeneráláshoz vagy gyermekmesék írásához, mivel hiányzik belőle a szöveg folytatásához szükséges generatív képesség.

Előnyei:

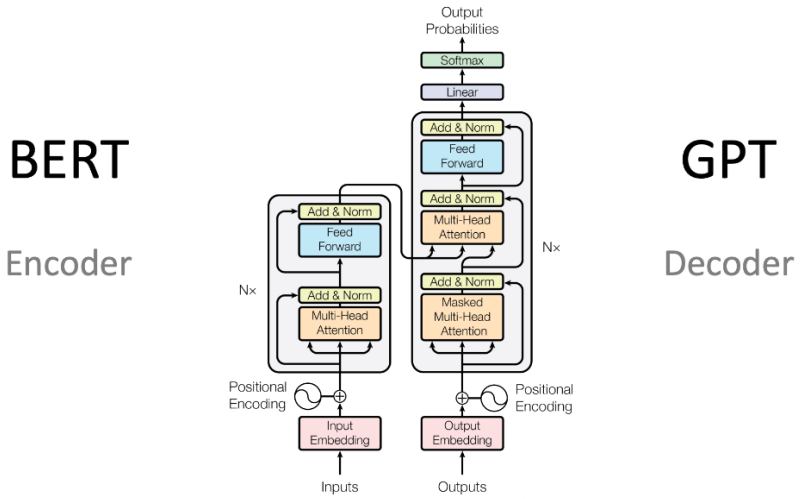
- Kiemelkedő szövegértési képesség: a kétirányú encoder révén a teljes kontextus figyelembevételével értelmezi a szöveget.
- Stabil és pontos, különösen szövegkiegészítési feladatokhoz, de nem rendelkezik dekóder elemmel a szövegalkotáshoz.

Hátrányai:

- Nem generatív, így hosszabb szövegek létrehozása korlátozott.
- A BERT csak kiegészítésre képes, nem tud teljes szöveget „feltalálni”, így nem felel meg a folyamatos történetgenerálás igényeinek.

12.3.1.1 BERT vs GPT

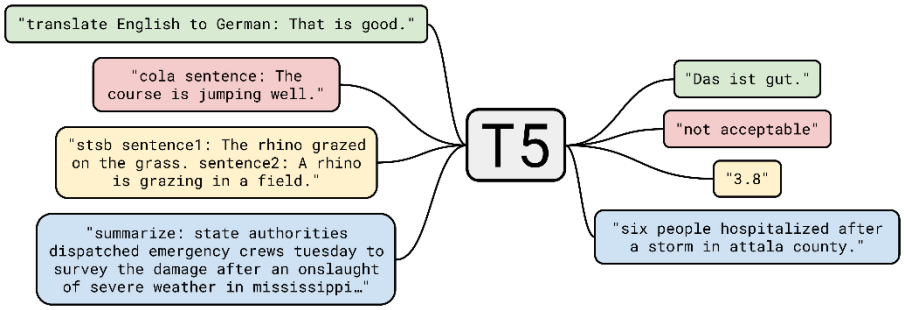
Míg a BERT modell hasonlít egy alaposan átvizsgáló és elemző „olvasóra”, amely megért minden részletet a szöveg mindkét irányából, a GPT modell egy **dekóder-alapú** rendszer, amely egy „mesélőhöz” hasonlítható. A GPT a szöveg kezdetétől haladva folytatja és generálja a történetet. Az encoder elemek helyett dekódert használ, ami lehetővé teszi számára, hogy önállóan új szöveget alkosson, és természetes módon tovább építse a már elkezdett mondatokat. Ezért a GPT kiválóan alkalmas kreatív szövegek, például történetek és mesék generálására, ahol a folytatás és a szövegalkotás kulcsfontosságú.



26. ábra BERT encoder - GPT decoder (Jung, 2024)

12.3.2 T5 (Text-To-Text Transfer Transformer)

A T5 egy rendkívül rugalmas modell, amely képes generatív feladatokat ellátni azáltal, hogy minden szövegfeldolgozási feladatot text-to-text (szövegből szöveg) problémaként kezel. Ez a rugalmasság ideális lehetne történetgenerálási célokra is, azonban a T5 modell komplexitása miatt jelentős számítási kapacitást igényel, és hosszabb szövegek generálása során időnként elveszíti a szöveg kohézióját. (Colin Raffel N. S., 2020)



27. ábra T5 modell *(Colin Raffel N. S., 2020)*

Előnyei:

Általános célú és rugalmas, sokféle feladathoz használható.

Képes egyszerűbb generatív feladatokra is, így akár mesék létrehozására is alkalmas lehet.

Hátrányai:

A T5 teljesítménye jelentős számítási erőforrást igényel, ami hátrányos lehet a kisebb projektek és otthoni fejlesztések számára.

Néha elveszíti a szöveg szerkezetét hosszabb generáció során, így a történetek folyamatos és koherens fenntartása nehéz lehet.

12.3.3 GPT-3

A GPT-3 a GPT-2 továbbfejlesztett, lényegesen nagyobb változata, amely jobb kontextus-fenntartási és történetgenerálási képességekkel rendelkezik. A GPT-3 alkalmasabb lehetett volna, mivel több paraméterrel rendelkezik, amelyek elősegítik a narratív koherenciát és a történetek hosszabb távú logikai folytonosságát. Azonban a GPT-3 használata nem csak technológiai szempontból, hanem anyagi és gyakorlati szempontból is kihívásokat jelent.

Előnyei:

Nagyobb paraméterszám, amely hosszabb és összetettebb történetek létrehozására alkalmas. Fejlett kontextus-megőrzési képességek, amelyek biztosítják, hogy a történet koherens maradjon akár hosszabb időn át is.

Hátrányai:

Jelentős számítási erőforrást igényel, amely hátrányos lehet kisebb projektek és személyes felhasználás esetén.

Költséges és korlátozott hozzáférés, mivel az felhőalapú platform keresztül érhető el.

12.3.4. A GPT-2 Előnyei a Fenti Modellekkel Szemben

A GPT-2 ideális kompromisszum a gyermekmese generátor létrehozása során, mivel

- **Megfelelő méretű modell:** A GPT-2 paramétereinek száma és architektúrája lehetővé teszi a hatékony generálást, mégis kezelhető számítási igényekkel rendelkezik.
- **Nyílt forráskódú és könnyen elérhető:** Mivel a GPT-2 modell nyílt forráskódú, így könnyen finomhangolható és módosítható a specifikus igényekhez.
- **Könnyen adaptálható gyermekbarát történetgenerálásra:** A modell különböző prompt-struktúrák alkalmazásával finomhangolható, és megfelelő mesebeli karakterek, helyszínek és események beillesztésével képes volt egyszerű, gyermekek számára érthető nyelvezetű történeteket generálni.

12.3.5 Összegzés

A GPT-2 kiválasztása egy megfontolt döntés volt, mivel a modell számos szempontból alkalmasnak bizonyult a mese generálására. A BERT és a T5 modellek ugyan jó alternatívák lehettek volna, de a generatív képességek hiánya és a számítási igények miatt nem feleltek meg teljesen a céloknak. A GPT-3 erősebb lehetett volna, de a korlátai miatt a GPT-2 maradt a legoptimálisabb választás.

12.4 Első Kísérlet: A „1000 Folk Stories Around the World” Adatkészlet

Az első modell fejlesztéséhez a Kaggle-ről letöltött “1000 Folk Stories Around the World” adatkészletet használtam, amely különböző népmeséket és folklórt tartalmazott a világ minden tájáról. Az adatokat a Kaggle API-n keresztül töltöttem le és egy `DataFrame`-be rendeztem:

```
import kagglehub
import pandas as pd
import os

# Adathalmaz letöltése
path = kagglehub.dataset_download("chayanonc/1000-folk-stories-around-the-world")

# Adatok betöltése DataFrame-be
df = pd.read_csv(os.path.join(path, os.listdir(path)[0]))

# Szövegek kinyerése
stories = df['full_text'].tolist()
```

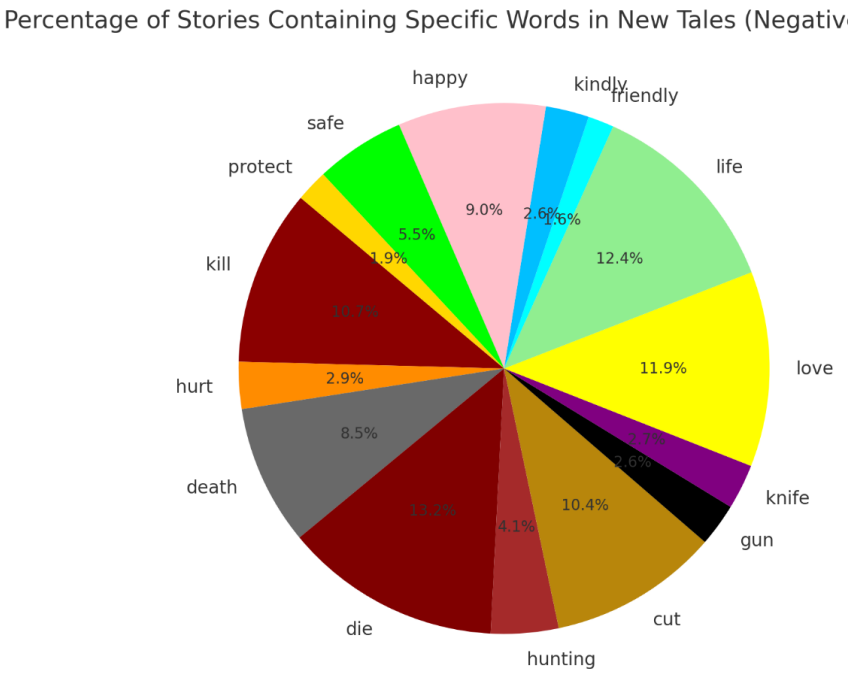
28. ábra Kaggle dataset kinyerése

12.4.1 Az Adatkészlet és annak korlátai

Ez az adatkészlet történeteket tartalmazott, amelyek eltérő kultúrákból és korosztályokból származtak, sok esetben felnőtteknek szánt történeteket is magában foglaltak. Mivel az eredeti adatbázis célközönsége vegyes volt, a tartalmak stílusa és témája nem volt konzisztens gyermekbarát történetek számára. Az adatstruktúra elemzése során egyértelművé vált, hogy a történetek gyakran:

- Erőszakos vagy brutális eseményeket tartalmaztak (például halálesetek, sebesülések, fenyegetések).
- Logikátlan cselekményvezetést mutattak, ami nehezítette a történetek követhetőségét.
- Komplex és sötét témákat öleltek fel, amelyek nem voltak megfelelőek gyermekek számára.

Az lenti ábra megmutatja egyes szavak gyakoriságát, amely azt ábrázolja, hogy a generált történetek hány százaléka tartalmazott negatív jelentéstartalmú szavakat, például „halál”, „ölni”, „fájdalom”. Ezeknek a szavaknak a gyakori előfordulása rávilágított arra, hogy az adatbázis nem volt alkalmas gyermekbarát történetek generálására.



29. ábra Szavak gyakoriságának százalékos eloszlása

12.4.2 Finomhangolási Beállítások és Eredmények

A modell finomhangolását követően azt tapasztaltam, hogy a generált történetek nem feleltek meg a gyermekbarát tartalmi elvárásoknak. Az alábbi kódrészlet mutatja a finomhangolási folyamat egyes paramétereit, amelyekkel próbáltam kontrollálni a generált történetek hosszát és variabilitását, de végül ezek sem eredményeztek kielégítő minőséget:

```
# Finomhangolási beállítások
training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=2,
    per_device_train_batch_size=1,
    per_device_eval_batch_size=1,
    warmup_steps=200,
    weight_decay=0.01,
    logging_dir='./logs',
    logging_steps=10,
)

# Trainer inicializálása
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=dataset,
)

# Modell betanítása
trainer.train()
```

30. ábra Első verzió finomhangolás paraméterek

12.4.3. Példák a Nem Megfelelően Generált Történetekre

Az alábbiakban néhány példát mutatok be a rosszul generált történetek közül, amelyek világosan szemléltetik a modell hibáit:

12.4.3.1 Brutális és zavaros cselekmény:

- **Generált történet:** „He went to the forest to see what was out there... they told him that his father had died and that their children were dead."
- **Probléma:** A történet logikátlan és sötét tónusú, ami nem alkalmas gyermekek számára

12.4.3.2 Érthetetlen cselekményvezetés:

- **Generált történet:** „He took his time to think, and he determined to try his luck... then he looked up and saw a small hole where the other two holes had been."
- **Probléma:** A történet nem követhető, nincsenek egyértelmű fordulópontok és lezárások, a cselekményvezetés kusza és értelmezhetetlen

12.4.3.3 Szélsőségesen agresszív karakterek:

- **Generált történet:** „Once upon a time, there was a joyful and adventurous child named Adam... He met a young man named Jai... who was very angry and threatened to kill everyone."
- **Probléma**: Az agresszív karakterek és a fenyegető jelenetek teljesen ellentétesek a gyermekbarát történetgenerálás elveivel

12.4.4 Miért Volt Szükséges a Modell Elvetése?

Az első modell elvetésére több tényező miatt volt szükség:

- **Adatforrás:** Az adatkészlet tartalma túl sokszínű és gyakran nem illeszkedett a gyermekmesék világához. Az eltérő kulturális és tematikai eredet miatt a modell képtelen volt konzisztens, gyermekeknek megfelelő narratívát kialakítani.
- **Negatív Tartalmak:** A szóhasználat és a cselekmények gyakran tartalmaztak negatív és erőszakos elemeket. A fent említett ábra is bizonyítja, hogy a történetek jelentős része tartalmazott nem kívánatos, ijesztő szavakat.
- **Történet Koherencia Hiánya:** Az adatbázis változatos történetstruktúrái miatt a modell sokszor logikátlan vagy követhetetlen történeteket generált. Ez különösen problémás volt a mese szempontjából, hiszen egy gyermekmese esetében alapvető elvárás a könnyen érthető, egyszerű szerkezet.

12.4.5 A Megoldás: Új Adatkészlet és Modell Finomhangolás

Miután nyilvánvalóvá váltak a fenti problémák, új megközelítést kellett alkalmaznom. Az új adatkészlet a “bedtime stories” datasetet használtam, amely gyermekek számára készült, egyszerű és követhető történeteket tartalmazott. Ezzel az új adatstruktúrával sokkal konzisztens és gyermekbarátibb történetek generálása vált lehetővé.

12.5 Az Új Adatkészlet

A mese generátor modell betanításához a Hugging Face nyílt forrású adatbázisát használtam, a **"gofilipa/bedtime_stories"** elnevezésű adatkészletet. Ez az adathalmaz gyerekmeséket tartalmaz, amelyek stílusában és nyelvezetében is alkalmasak a kívánt történetgenerálásra, hiszen ezek a történetek egyszerű, gyermekek számára érthető nyelvet használnak.

„ https://huggingface.co/datasets/gofilipa/bedtime_stories „

12.5.1 Az előkészítés

A stories oszlopból kinyert szövegeket egy listába rendeztem, ami az alapját képezte a finomhangolás során használt tanítóadatoknak. Az adatok feldolgozása során biztosítottam, hogy az alapvető szövegstruktúrák és a történetek egyszerűsége megmaradjon, mivel ez kulcsfontosságú volt a célközönség (gyermekek) szempontjából.

12.5.2 Tokenizálás

Az adatok feldolgozása során a GPT-2 modellhez igazítottam a szöveget. A tokenizálás a szövegfeldolgozás első lépése, ami a mondatok szavakra és karakterekre bontását jelenti. Itt a GPT2Tokenizer használatával a modell képes volt a mese-adatok megfelelő strukturálására.

Ezt követően egy saját adatstruktúrát hoztam létre a CustomDataset osztályban, amely a tokenizált szövegeket tárolja és előkészíti a modell betanításához. Ezzel biztosítottam, hogy a teljes szövegkorpusz megfelelően felhasználható legyen az NLP folyamatokhoz, különösen a történetek szekvenciális adatfeldolgozása szempontjából:

```
# Adathalmaz betöltése a Hugging Face-ről
df = pd.read_csv("hf://datasets/gofilipa/bedtime_stories/stories.csv")

# Ellenőrizzük az oszlopneveket
print(df.columns)
```

```
Index(['stories'], dtype='object')
```

```
# Szövegek kinyerése a 'stories' oszlopból
stories = df['stories'].tolist()

print(stories[:1])  # Az első mese kiíratása
```

```
['The stars twinkled in the night sky as little Eva lay in her bed dreaming of adventures. She imagin
```

```
# Modell és tokenizer betöltése
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
tokenizer.pad_token = tokenizer.eos_token
model = GPT2LMHeadModel.from_pretrained('gpt2')

# Eszközbeállítás
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)

# Adatok tokenizálása
inputs = tokenizer(stories, return_tensors='pt', max_length=512, truncation=True, padding=True)

# Dataset létrehozása
class CustomDataset(torch.utils.data.Dataset):
    def __init__(self, encodings):
        self.encodings = encodings

    def __getitem__(self, idx):
        item = {key: torch.tensor(val[idx]) for key, val in self.encodings.items()}
        item['labels'] = item['input_ids'].clone()
        return item

    def __len__(self):
        return len(self.encodings['input_ids'])

dataset = CustomDataset(inputs)
```

31. ábra Tokenizálás

12.5.3 Finomhangolás

A modell teljesítményének optimalizálása és a kívánt eredmények elérése érdekében különböző finomhangolási paramétereket alkalmaztam, amelyek segítségével a modell folyamatosan javíthatta a generált tartalmak minőségét. Az alábbiakban részletesen bemutatom a finomhangolás során beállított paramétereket és a tréning folyamat főbb eredményeit.

A mese generátor modell finomhangolásának beállításai kritikusak voltak, mivel a generált szövegeknek érthetők, strukturáltak és gyermekbarátoknak kellett lenniük. A modell konfigurációja a következő kulcspontok mentén zajlott:

```
# Finomhangolási beállítások
training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=3,
    per_device_train_batch_size=2,
    warmup_steps=200,
    weight_decay=0.01,
    logging_dir='./logs',
    logging_steps=50,
)

# Trainer inicializálása
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=dataset,
)

# Modell betanítása
trainer.train()
```

32. ábra Finomhangolás paraméterei

12.5.3.1 Finomhangolási Paraméterek

A TrainingArguments konfigurációban megadott paraméterek a finomhangolási folyamat főbb jellemzőit határozták meg, ezek közül a legfontosabbak:

- **output_dir='./results':** A modell tréningje során keletkező eredmények és állapotmentések tárolási útvonala. Ez lehetővé tette, hogy a betanított modellt bármikor visszaállítsam és újra betöltsem.
- **num_train_epochs=3:** A betanítási ciklusok (epoch-ok) száma, azaz hány alkalommal haladt végig a modell az adathalmazon. A 3 epoch megfelelő egyensúlyt biztosított a modell teljesítménye és a túlilleszkedés elkerülése között. Ennél több epoch használata növelhette volna a teljesítményt, de növelte volna a túlilleszkedés kockázatát.
- **per_device_train_batch_size=2:** Az egyes eszközönként (például GPU-nként) betöltött minták száma egy betanítási iteráció során. A kis batch size érték részletesebb, finomabb modellezést tett lehetővé, bár lassabb tréningfolyamathoz vezetett. Az adatmennyiség korlátai miatt ez az érték optimálisnak bizonyult.
- **warmup_steps=200:** A betanítás kezdeti szakaszában alkalmazott „felvezetési lépések” száma. Ez lehetővé tette, hogy a modell lassan kezdje el a tanulást, fokozatosan növelve a tanulási rátát, így a kezdeti fázisban elkerülve a gyors és instabil változásokat a súlyokban.
- **weight_decay=0.01:** A súlycsökkenési paraméter hozzájárult a modell generalizációs képességének javításához. Az érték megfelelő egyensúlyt teremtett az illeszkedés és a túlilleszkedés között, így csökkentve annak esélyét, hogy a modell a betanító adatok egyedi mintázataira túlzottan ráálljon.
- **logging_dir='./logs' és logging_steps=50**: Az edzés során minden 50. lépésnél naplózás történt, amely lehetővé tette a tanulási folyamat részletes nyomon követését és a szükséges beavatkozások elvégzését. A naplófájlok segítségével a finomhangolási folyamat során fellépő esetleges problémák azonosítása gyorsan és hatékonyan megoldhatóvá vált.

12.5.3.2 Finomhangolás Folyamat és Eredmények

A tréning folyamat kezdeti szakaszaiban a modell viszonylag magas veszteségi értékkel (training loss) kezdett, amely fokozatosan csökkent az epochok előrehaladtával. Az alábbi táblázat szemlélteti a tanulási folyamat során rögzített veszteségi értékeket:

Step	Training Loss
50	3.227300
100	1.927700
150	1.721300
200	1.576400
250	1.417200
300	1.426400

33. ábra Finomhangolás eredmények

Az eredmények alapján a modell tréning loss értéke az első 200 lépés során jelentősen csökkent, jelezve, hogy a modell egyre pontosabban illeszkedett az adathalmazra. Az 1.5 körüli loss érték elérése után a javulás üteme lassult, ami arra utal, hogy a modell közeledett az optimális beállításhoz. A teljes tréningfolyamatot az alábbi összegzett eredmények foglalják össze:

- Global step: 300
- Training loss: 1.8827
- Runtime: 492.5 másodperc
- Train samples per second: 1.212
- Train steps per second: 0.609
- Total FLOPS: 45,091,247,616,000 (Floating Point Operations per Second)

Az epoch-ok során elért 1.88-as átlagos training loss érték megfelelően alacsonynak bizonyult, ami jelezte, hogy a modell sikeresen megtanulta az adatokból a mintázatokat anélkül, hogy túlságosan az adathalmaz specifikus elemeire összpontosított volna. Az eredmények alapján a finomhangolt modell képes volt stabil, konzisztens teljesítményt nyújtani a tesztelési folyamat során, jelezve, hogy a beállítások optimálisak voltak a gyermekbarát mese generátor létrehozásához.

12.5.3.3 Összegzés

A finomhangolási paraméterek megfelelő megválasztása alapvető fontosságú volt a modell sikeres működése szempontjából. Az alacsony batch size, az optimális epoch-szám és a súlycsökkenés hozzájárultak a modell generálási képességének növeléséhez, amely elengedhetetlen a gyermekbarát mesék generálásához.

Az alkalmazott finomhangolási beállításoknak köszönhetően a modell megbízhatóan generált gyermekbarát meséket, amelyek érthetőek, következetesek és kreatívak voltak. Később majd a két különböző prompt-alapú megközelítés eredményeinek részletes elemzését és összehasonlítását mutatom be, amelyek különféle struktúrákat használtak.

12.5.4 Generált szöveg paraméterezése

A GPT-2 modell konfigurációja lehetővé teszi olyan paraméterek beállítását, mint a temperature (hőmérséklet) és top_k vagy top_p paraméterek, amelyek befolyásolják a generált szöveg kreativitását és újszerűségét. Ez kiemelten fontos a mese generálásnál, mivel ezek a paraméterek segítik az egyes történetek érzelmi tónusának és összetettségének finomhangolását.

Például alacsonyabb hőmérséklet esetén a modell konzervatívabban választ új szavakat, amely stabilabb, gyermekbarát szöveget eredményez, míg magasabb hőmérséklet nagyobb varianciát és kreatívabb tartalmat biztosít. Az alábbi példa egy közepes, 0.7-es hőmérséklet beállítását mutatja, ami megfelelő egy gyermekek számára érthető és változatos történet létrehozására:

```python
# Minimális paraméterekkel működő történetgeneráló funkció
def test_fine_tuned_model_simple(protagonist="A brave adventurer Adam", max_length=1000):
    # Egyszerű, mesés bevezető prompt
    intro_prompt = f"Once upon a time, {protagonist} went on an incredible journey."

    # Prompt tokenizálása és átküldése a modellnek
    input_ids = tokenizer.encode(intro_prompt, return_tensors='pt').to(device)

    # Generálás a modell segítségével
    output = model.generate(
        input_ids,
        max_length=max_length,
        num_return_sequences=1,
        no_repeat_ngram_size=2,
        temperature=0.6,  # kreativitás
        top_k=90,         # irányítottabb generálásért
        top_p=0.95,       # összpontosított történetért
        do_sample=True,
        pad_token_id=tokenizer.eos_token_id
    )

    # Kimenet dekódolása és visszaadása
    story = tokenizer.decode(output[0], skip_special_tokens=True)
    return story

# Teszt futtatása egy alapértelmezett főszereplővel
generated_story = test_fine_tuned_model_simple()

print("Generated Story:")
print(generated_story)
```

34. ábra Generálási paramtéerek

Ezek a beállítások lehetővé tették, hogy a történetfolyam stabil maradjon, de kellően változatos legyen a gyermekek érdeklődésének fenntartása érdekében.

12.5.5 Modellek Tesztelése

A mese generátor modell tesztelésénél két eltérő megközelítést alkalmaztam a történetek felépítésére. Az első modell egy részletesen strukturált prompt rendszeren alapult, amely egy kezdő promptból, több bekezdésen át ismétlődő középső promptokból és egy záró promptból állt. A második modell ezzel szemben egy minimalista prompt struktúrára épült, amelyben a történet kizárólag egy rövid bevezető prompt alapján indult, majd ezt követően a modell saját maga alakította a történet menetét. A cél mindkét esetben egy gyermekbarát, koherens mese generálása volt.

12.5.5.1 A Részletesen Strukturált Prompt Rendszer Tesztelése

A részletesen strukturált modell minden szakaszához külön promptokat határoztam meg, amelyekkel irányítottam a cselekményvezetést és a karakterek cselekedeteit. Az alábbi példa kódrészlet bemutatja ezt a struktúrát:

```
# Történetgeneráló funkció a finomhangolt nyelvezettel és stílusban
def generate_story(protagonist, location, brief_event, paragraphs, words_per_paragraph):
    # Alap prompt a megadott paraméterekkel, a finomhangolt nyelvezet alapján
    initial_prompt = (f"In the {location}, a brave explorer named {protagonist} found themselves on an unusual adventure. "
                      f"One day, {protagonist} discovered {brief_event}. This discovery would lead to wonders and challenges beyond their wildest dreams.")

    story = ""
    prompt = initial_prompt  # Start with the initial prompt for the first paragraph

    for i in range(paragraphs):
        # Generate input_ids with attention mask and set pad token id for reliable generation
        input_ids = tokenizer.encode(prompt, return_tensors='pt').to(device)
        attention_mask = torch.ones(input_ids.shape, device=device)

        output = model.generate(
            input_ids,
            max_length=words_per_paragraph + len(input_ids[0]),
            num_return_sequences=1,
            no_repeat_ngram_size=2,
            temperature=0.7,  # Reduced temperature for more controlled output
            top_k=30,  # Lowered top_k for more focused choices
            top_p=0.9,
            do_sample=True,
            attention_mask=attention_mask,
            pad_token_id=tokenizer.eos_token_id
        )

        paragraph = tokenizer.decode(output[0], skip_special_tokens=True)

        # Add the generated paragraph to the story, removing any repeated prompt text
        story += "\n\n" + paragraph[len(prompt):]

        # Reset the prompt to keep reinforcing the initial setup for each new paragraph
        prompt = (f"As {protagonist} continued their adventure in the {location}, "
                  f"they thought back to the moment they discovered {brief_event}. Each step was filled with new mysteries and friends along the way.")

    return story.strip()

# Teszt mese generálása a megadott paraméterekkel
test_story = generate_story(
    protagonist="Adam",
    location="enchanted forest",
    brief_event="a glowing, magical animal",
    paragraphs=3,
    words_per_paragraph=100
)

print("Generated Test Story:")
print(test_story)
```

35. ábra Struktúrált Prompt

12.5.5.2 Egyszerűbb Prompt: Minimális Kezdeti Prompttal Irányított Történet

A következő megközelítés során egy egyszerűbb struktúrát alkalmaztam, amelyben a történet egy rövid bevezető prompt alapján indult, és onnantól teljesen a modell vezette a cselekmény irányát. Ez a felépítés biztosította a történet szabadabb fejlődését, amivel változatosabb és természetesebb történetek születtek.

```python
# Minimális paraméterekkel működő történetgeneráló funkció
def test_fine_tuned_model_simple(protagonist="A brave adventurer Adam", max_length=300):
    # Egyszerű, mesés bevezető prompt
    intro_prompt = f"Once upon a time, {protagonist} went on an incredible journey."

    # Prompt tokenizálása és átküldése a modellnek
    input_ids = tokenizer.encode(intro_prompt, return_tensors='pt').to(device)

    # Generálás a modell segítségével
    output = model.generate(
        input_ids,
        max_length=max_length,
        num_return_sequences=1,
        no_repeat_ngram_size=2,
        temperature=0.6,  # kreativitás
        top_k=90,         # irányítottabb generálásért
        top_p=0.95,       # összpontosított történetért
        do_sample=True,
        pad_token_id=tokenizer.eos_token_id
    )

    # Kimenet dekódolása és visszaadása
    story = tokenizer.decode(output[0], skip_special_tokens=True)
    return story

# Teszt futtatása egy alapértelmezett főszereplővel
generated_story = test_fine_tuned_model_simple()

print("Generated Story:")
print(generated_story)
```

36. ábra Minimális promptal

12.5.6 Eredmények

A tesztelési eredmények elemzése során kiderült, hogy a két megközelítés jelentős különbségeket mutatott a történetek minőségében, áramlásában és kreativitásában. A részletesen strukturált prompt rendszer nem váltotta be a hozzá fűzött reményeket, míg az egyszerűbb prompt-tal vezérelt történetgenerálás kedvezőbb eredményekhez vezetett.

12.5.6.1 Részletes Prompt Rendszer: Hátrányok és Problémák

A részletes prompt struktúrával a generált történetek az alábbi problémákkal szembesültek:

- **Túlzott Ismétlések:** Az ismételt középső promptok miatt a történet sokszor redundáns kifejezéseket és cselekményelemeket tartalmazott, ami monotonná tette a narratívát.
- **Kevésbé Természetes Áramlás:** A különálló promptok hatására a történet mesterséges hatást keltett, amely zavaró lehetett a mese olvasása közben. A történet kevésbé tűnt folyékonynak,és nem volt harmonikus az átmenet a cselekmények között.
- **Túlzott Irányítás:** Mivel minden egyes szakaszt előre meghatározott promptok vezettek, a modell kevés teret kapott arra, hogy szabadon irányítsa a cselekményt. Ezáltal a történetek kevésbé lettek kreatívak és váratlanok.

12.5.6.2 Példák a Részletesebb promptal generált mesékre

1.verzió

> He soon became a fast traveler, visiting different lands and meeting new friends. Soon, they had a cozy cozy bed, and soon they were ready to explore the world together. But when they finally reached the bottom of the forest they saw a friendly little owl, who thanked them for their good fortune and promised them a happy day ahead.
>
> One night, a friend asked for a little star to light up a forest. The star lit up the night sky and they all got a surprise. They found a new friend and their little rabbit friend, who loved the magical creatures. With a smile on their faces, the rabbits and the bunny got back home and fell asleep.

3.verzió

> In a magical place called enchanted forest, there lived a brave soul named Adam. One day, Adam discovered a glowing, magical animal, which marked the beginning of an incredible journey.
>
> With each step in the enchanted forest, Adam encountered many wonders and challenges. They couldn't help but think about a glowing, magical animal as they continued their journey. Eventually, they met a little owl named Lola who told them stories about how they lived in a magical forest. Lila then flew away with them to a peaceful village where they discovered a fire and a pond. From then on, the two of them would explore the world together, laughing and singing, until it was time for bed.

With each step in the enchanted forest, Adam encountered many wonders and challenges. They couldn't help but think about a glowing, magical animal as they continued their journey. One day, a little bunny hopped around the forest and saw a magical pond. He asked it to join them in a pond and they both jumped out. With a hug, the bunny thanked the animal for all the kindness and said, "I wish you a happy day tomorrow."

As the adventure came to a close, Adam looked back on their journey in the enchanted forest. They felt changed by the experience of a glowing, magical animal and knew they would never forget it. But soon, the sun began to set and the stars began coming out of their windows. As the moon fell, they watched as the magical creatures flew away, dreaming of the exciting adventures they'd have the next day.

12.5.6.3 Egyszerűbb Prompt Rendszer: Előnyök és Pozitív Eredmények

Az egyszerűbb prompt rendszer több szempontból is sikeresebbnek bizonyult:

- **Természetesebb és Folyékonyabb Narratíva:** A kevesebb prompt használata lehetővé tette, hogy a történet saját áramlását kövesse, ami sokkal természetesebb, harmonikusabb történetvezetést eredményezett.
- **Kreativitás és Variabilitás:** A modell a minimalista promptnak köszönhetően nagyobb szabadságot kapott a cselekmény vezetésében, ami kreatívabb, sokszínűbb meséket eredményezett, anélkül, hogy az ismétlődés vagy a redundancia megjelenne.
- **Kedves és Egyszerű Hangnem:** A rövidebb prompt hatására a történetek barátságosabbá és egyszerűbbé váltak, ami ideális a gyermekek számára. A 0.1, 0.2 és 0.4 verziók különösen pozitív visszajelzéseket kaptak a történetek kedves és varázslatos hangulata miatt.

12.5.6.4 Példák az Egyszerűbb promptolással generált mesékre

0.1 Verzió: Ez a verzió rövid, kedves és könnyen követhető, varázslatos kalandokat mutat be.

„Once upon a time, A brave adventurer went on an incredible journey. He explored the world, meeting new people and discovering new lands. But when he reached the magical kingdom of Amethyst, he was met by a mysterious dragon. With a special gift from his new friend, the dragon granted him the power to fly and explore. Together, they flew around the kingdom, exploring all kinds of wondrous places. Eventually, their journey was over and they returned home safely to their homes.”

0.2 Verzió: Ebben a verzióban a történet természetes áramlása segít a barátság és elismerés témájának megjelenítésében.

> „Once upon a time, A brave adventurer Adam went on an incredible journey. He explored the world, meeting new creatures, and meeting interesting people. From his adventure, he was able to make friends with the most amazing animals on the planet. In the end, Adam was named the best adventurer in the universe."

0.4 Verzió: Ez a történet különösen jól mutatja, hogyan képes a modell önállóan egy kedves, gyermekbarát történetet generálni

> „Once upon a time, A brave adventurer Adam went on an incredible journey. He flew across the sky and met many beautiful animals. One night, he fell asleep and found himself in a magical forest. After a while, the animals began to play music, singing, dancing and dancing until he finally fell into a peaceful sleep."

12.5.6 Következtetések

Az eredmények alapján egyértelművé vált, hogy a minimalista prompt-rendszer a legalkalmasabb megoldás a gyermekbarát mese generálásához, mivel lehetőséget biztosít a modell számára, hogy önállóan vezesse a történetet. Az egyszerű promptokkal irányított struktúra könnyed, folyamatos történeteket eredményezett, amelyek természetes áramlása harmonikusabb és folyékonyabb volt, és ez különösen kedvezőnek bizonyult a gyermekmesék esetében.

A részletes prompt rendszer bár irányítást biztosított, túlságosan megkötötte a modellt, így a történetek sokszor mesterségesnek, ismétlődőnek és kevésbé kreatívnak bizonyultak. A minimalista prompt használatával a történetek sokkal színesebbé, élvezetesebbé és gyermekbaráttá váltak, mivel a modell képes volt saját belátása szerint alakítani a cselekményt, ami új és váratlan eseményekkel gazdagította a meséket.

12.5.7 Javaslatok

1. További Finomhangolást: A modell finomhangolása során a hőmérséklet és a top_k/top_p értékek további optimalizálása még inkább javíthatja a történetek minőségét.

3. Variábilis Promptok Kialakítását: Olyan prompt-struktúrát hozzunk létre, amely rugalmasan alkalmazkodik a különböző bevezető szövegekhez, és változatosan indítja el a történetet. Ez azt jelenti, hogy a modellnek nem mindig ugyanazt a bevezetést vagy kezdő mondatot adjuk meg, hanem különféle indítási lehetőségeket kínálunk számára

13. ÖSSZEGZÉS

Az NLP célja az emberi nyelv gépi megértése és feldolgozása, amely számos ágazat számára kínál új automatizálási lehetőségeket, különösen a szövegfeldolgozás, gépi fordítás, szövegértés, és szöveggenerálás területein. A dolgozat áttekintette az NLP főbb fejlődési szakaszait és bemutatta a transzfer tanulás jelentőségét az előre betanított, nagy nyelvi modellek finomhangolásában.

13.1 Elméleti Keretek és Transzfer Tanulás Alkalmazása

A transzfer tanulás különösen előnyösnek bizonyult az NLP-ben, mivel lehetőséget biztosít arra, hogy a nagyméretű nyelvi modellek által megszerzett általános nyelvi tudást speciális feladatokra adaptáljuk. A dolgozatban ismertetett modellek, mint a BERT, GPT-2 és T5, alkalmasak különböző nyelvi feladatok ellátására, amelyek különösen relevánsak a szöveganalízis és szövegalkotás során. Az NLP területén folytatott jelenlegi kutatások középpontjában a még pontosabb nyelvi reprezentációk, a finomhangolási technikák optimalizálása és a hatékonyabb prompt-tervezés áll. A kutatások egyre inkább arra összpontosítanak, hogy a nyelvi modellek érzékenyek legyenek a kontextusra, képesek legyenek az emberi interakciókat is figyelembe venni, valamint kezeljék a különböző nyelvek és dialektusok közötti árnyalatokat.

13.2 Esettanulmány: Gyermekbarát Történetgenerátor és Tanulságai

A dolgozat részeként bemutatott esettanulmányban egy gyermekbarát mese generátor modell fejlesztését végeztem el, amely a GPT-2 modellre épült. Az esettanulmány célja a transzfer tanulás gyakorlati hasznosításának bemutatása volt egy specifikus adatset alapján. A modell finomhangolása különböző adatforrások használatával zajlott, ahol a megfelelő adatkészlet kiválasztása, a prompt-tervezés és a finomhangolás optimalizálása kritikus fontosságú volt a kívánt eredmények elérése érdekében.

A kezdeti adatbázis – amely változatos népmeséket tartalmazott – nem bizonyult megfelelőnek a gyerekek számára szánt történetek generálásához, mivel gyakran tartalmazott agresszív, sötét vagy logikátlan elemeket. Az adatkészlet cseréje, a hugging face-ről a „bedtime stories” adathalmaz használata lehetővé tette, hogy a modell egyszerű, gyermekek számára kedves és érthető történeteket generáljon. Az esettanulmány során alkalmazott minimalista prompt-struktúra biztosította, hogy a modell természetes, áramló szöveget generáljon, amely különösen kedvező a gyermekbarát tartalmak előállításához.

13.3 Jelenlegi Kihívások és jövőbeli kilátások az NLP és a Transzfer tanulás területén

A dolgozat eredményei alapján a transzfer tanulás rendkívül hatékony eszköz az NLP-ben, de a technológia jelenlegi formájában is számos kihívással néz szembe. Az egyik jelentős probléma a nagy nyelvi modellek adatintenzitása és számítási igénye, amely jelentős erőforrásokat igényel. A jövőbeli kutatások fontos célkitűzése ennek a számítási igénynek a csökkentése, valamint az energiahatékonyabb modellarchitektúrák fejlesztése. Az optimalizált modellméretek, mint a distillált változatok (pl. DistilBERT), lehetőséget nyújtanak a kisebb, mégis hatékony modellek létrehozására, amelyek nemcsak költséghatékonyabbak, hanem környezetbarátabbak is.

Egy másik kihívás a generált szövegek tartalmának kontrollálása és az etikai szempontok figyelembevétele. A generált szövegek elfogultsága, illetve az érzékeny témákban történő alkalmazás etikai kérdéseket vet fel. Ezek problémák megoldása érdekében egyre több kutatás foglalkozik az NLP modellek „megbízhatóságának” fejlesztésével, a tartalom szűrésével és a modellek tudatos finomhangolásával, hogy elkerüljük az érzékeny vagy nemkívánatos tartalmak előállítását.

13.4 Javaslatok a Jövőbeni Fejlesztésekre

A transzfer tanulás és az NLP területén a következő fejlesztési irányok lehetnek meghatározóak:

1. Adaptív és Etikus Modellek Fejlesztése: Az NLP modellek adaptívabbá tétele a különböző nyelvi és kulturális kontextusokhoz az egyik legnagyobb kihívás. A jövőbeli kutatások célja olyan modellek létrehozása, amelyek képesek figyelembe venni a kulturális sajátosságokat, és minimalizálni a sztereotípiákat és elfogultságokat.

2. Finomhangolás Hatékonyságának Növelése: A finomhangolási technikák fejlesztése, különösen a kisebb adatkészleteken való alkalmazhatóság növelése szintén kiemelt kutatási irány. Az optimalizált finomhangolási megközelítések célja, hogy a modellek pontosan és hatékonyan alkalmazkodjanak az új feladatokhoz, még akkor is, ha csak korlátozott mennyiségű adat áll rendelkezésre.

3. Energiahatékony Modellek és Green NLP: Az NLP területén egyre nagyobb hangsúlyt kap a „Green NLP”, amely az energiahatékony modellek és algoritmusok fejlesztését célozza. Az energiahatékonyságra való törekvés nemcsak gazdasági előnyökkel jár, hanem a fenntarthatóság szempontjából is kulcsfontosságú.

14. IRODALOMJEGYZÉK

Ashish Vaswani, N. S. (2023, Aug 2). *Cornell University*. Retrieved from Attention Is All You Need: https://arxiv.org/abs/1706.03762

Binxuan Huang, K. M. (2019). Syntax-Aware Aspect Level Sentiment Classification with Graph Attention Networks. Carnegie Mellon University.

ColdSceptical. (2024. April 4). Emerging Trends in the Future of Natural Language Processing.

Colin Raffel, N. S. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research 21 (2020) 1-67*.

Colin Raffel, N. S. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Colin Raffel, Noam Shazeer, Adam Roberts ...*, 1-67. Retrieved from paperswithcode: https://paperswithcode.com/method/t5

Compressor, I. N. (2023, Oct 16). Retrieved from Medium: https://medium.com/intel-analytics-software/reduce-large-language-model-carbon-footprint-with-intel-neural-compressor-and-intel-extension-for-dfadec3af76a

deeplearning.ai. (2023. Jan 11). *Natural Language Processing*. Forrás: deeplearning.ai: https://www.deeplearning.ai/resources/natural-language-processing/

Durai, K. (2021, Jul 06). *medium*. Retrieved from Medium: https://medium.com/geekculture/basics-of-natural-language-processing-for-beginners-d86351df9d09

Emily M. Bender, T. G.-M. (2021. March). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . USA. Forrás: https://sci-hub.se/10.1145/3442188.3445922

Emma Strubell, A. G. (2019). Energy and Policy Considerations for Deep Learning in NLP. Cornell University.

Jacob Devlin, M.-W. C. (2019. May 24). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. Forrás: https://arxiv.org/abs/1810.04805

Jung, H. S. (2024, Aug 23). *학습과, 가장 성공적인 트랜스포머의 변형: BERT와 GPT 소개. 자기지도*. Retrieved from https://www.google.com/imgres?imgurl=https%3A%2F%2Fmiro.medium.com%2Fv2%2Fresize%3Afit%3A434%2F1*D5xg0yz7YzBSzS_F1efLAA.png&tbnid=dK3wCyu3ZF3KTM&vet=10CAYQxiAoA2oXChMIqJmhy_fAiQMVAAAAAB0AAAAAEFY..i&imgrefurl=https%3A%2F%2Fmedium.com%2F%40hugmanskj%2F%25EA

Marco Tulio Ribeiro, S. S. (2016, Aug 13). *ACM Digital Library*. Retrieved from https://dl.acm.org/doi/10.1145/2939672.2939778

Péter, D. R. (2018). Korszerű adatelemző algoritmusok alkalmazása a közigazgatásban. In D. R. Péter, *Adatelemzés könyv.* Budapest.

Scott M. Lundberg, S.-I. L. (2017). AUnified Approach to Interpreting Model Predictions. In *31st Conference on Neural Information Processing Systems.* Long Beach.

Shaikh, J. (2017. July 24). *medium*. Forrás: Medium: https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a

Shaikh, R. (2018. Oct 20). *medium*. Forrás: Medium: https://towardsdatascience.com/gentle-start-to-natural-language-processing-using-python-6e46c07addf3

Singh, V. (2024. Aug 15). *LLM Architectures Explained: NLP Fundamentals (Part 1)*. Forrás: Medium: https://medium.com/p/de5bf75e553a

Staff, N. (2020, Feb 12). Artificial Intelligence Expedites Brain Tumor Diagnosis during Surgery. Retrieved from Artificial Intelligence Expedites Brain Tumor Diagnosis during Surgery

Zoltán, P. B. (2024). Rendszerfejlesztés Órai anyag ppt. Budapest.

15. EGYÉB FORRÁSOK

Szakdolgozatomban felhasználtam az órai anyagokat:

- Órai anyag Adatbányászat – Dr. Racskó Péter
- Órai anyag Rendszerfejlesztés – Balogh Zoltán PhD.

A kódlista megtalálható az alábbi Google Colab linken:

https://colab.research.google.com/drive/14ASXuBNdUv76EAdeM-u2XfUCk3BmE0lM?usp=sharing

ClassX AI Children's Story Generator

https://www.kaggle.com/datasets/chayanonc/1000-folk-stories-around-the-world

https://huggingface.co/datasets/gofilipa/bedtime_stories

1. SZ. MELLÉKLET: ÁBRÁK JEGYZÉKE

FÜGGELÉKEK