SZÖVEGOSZTÁLYOZÓ MÓDSZEREK VIZSGÁLATA ÜGYFÉLSZOLGÁLATI KÉRÉSEK ELŐFELDOLGOZÁSÁHOZ

CSÉPÁNYI-FÜRJES LÁSZLÓ

A gyakori üzleti folyamatok feltárásához szükséges az eseménynaplók elemzésén túl más ügyfélkapcsolati csatornák tanulmányozása is. Ezen csatornák egyik legfontosabb eleme az e-mail. Mivel az e-mailek úgynevezett strukturálatlan szöveges formában léteznek, ezért kézenfekvő, hogy NLP-/NLU-technikákat vessünk be az üzleti folyamatok azonosítására, illetve feltárására. A kutatási modul célja, hogy az NLP-/NLU-szöveg tartalomosztályozási módszereit áttekintse és javaslatot tegyen egy szabad szöveges előfeldolgozó rendszer kidolgozására. Az ügyfelektől származó megkeresések előfeldolgozásával az ügyintéző munkája hatékonyabbá tehető, több idő szánható magára az ügyféllel végzett kommunikációra, illetve magára a feladat elvégzésére.

1. A kutatás célja és lépései

Az kutatás egy olyan koncepció kidolgozására irányult, mely alap lehet egy magyar nyelvű szövegosztályozó rendszer felépítéséhez. Ez a tervben megfogalmazott minimumcél eléréséhez volt szükséges. Egy konkrét nyelvi modell, a RoBERTa (Robustly optimized Bidirectional Encoder Representations from Transformers) alkalmazhatóságát jártuk körbe, mivel a szakirodalom tanulmányozása után ez tűnt a kézenfekvő választásnak, tekintve a rendelkezésünkre álló nyelvi korpuszok korlátozott terjedelmét, valamint azt a számítási teljesítményt, amit a kutatás során felhasználhatunk.

Technikai szempontból a cél egy olyan rendszer kidolgozása, mely képes magyar nyelvű strukturálatlan inputból feltárni az ügyfél lehetséges szándékait, ezen szándékokat pedig megfelelő folyamatazonosítókhoz képes rendelni.

- Minimumcél, hogy az inputot egy konkrét szándékfeltárás (intent detection) segítségével egy folyamat-azonosítóhoz kapcsoljuk.
- Maximumcél, hogy az inputból többszörös szándékot tárjunk fel (multi intent detection), valamint az egyes szándékokhoz kapcsolódó információkat (nevek, azonosítók, dátumok stb.) is kinyerjük (slot labeling).
- Célként megfogalmazódott, hogy doménspecifikus annotált tanítóminta hiányára is dolgozzunk ki egy eljárást. Tegyünk javaslatot, hogyan tudja az üzleti partner ezen tanítómintát saját hatáskörében, saját erőforrásának bevonásával elkészíteni.

1.1. Alapmodell létrehozása

A következő konkrét vizsgálatokat végeztük el, melyek során arra kerestük a választ, hogy alkalmas-e, és ha igen milyen formában az egyik legjelentősebb magyar nyelvű annotált korpusz, a Szeged Dependency Treebank (SZDT) arra, hogy segítségével szándékdetektálásra alkalmas nyelvi modellt hozzunk létre.

- Az elemzés során egy 10 000 mondatos alkorpuszt hoztunk létre (SZDT–10000) és az annotált formátumot annotálatlan szöveges formátumra alakítottuk, valamint elkészítettünk egy társállományt, mely a mondatok függőségi nyelvtanból ismert ún. gyökér (ROOT) fogalmát tartalmazza. Ezen két állomány segítségével szándékmegállapítás-kísérletet lehet végezni, mégpedig élve azzal a feltételezéssel, hogy a mondat a gyökér szava által definiált szándékot indukálja.
- Egy még szűkebb, 7000 mondatos alkorpuszon hoztunk létre egy előtanított (pre-trained) RoBERTa modellt, melynek az SZDTROBERTA–7000 munkanevet adtuk.
- Elvégeztük az elkészült modell szövegosztályozásra történő finomhangolását és kiértékelését.
- Ugyanezen kísérletet az angol nyelvű, NLU-BENCHMARK tanítómintán is elvégeztük, a ROBERTA-BASE előtanított modell felhasználásával.
- A Nyelvtudományi Intézet HIL_ROBERTA modelljét is bevontuk a vizsgálatokba arra keresvén a választ, hogy a mi előtanított modellünk versenyképes megoldást nyújt-e.

A kísérletekhez a PyTorch-könyvtárat használtuk. Egy Nvidia GTX 1050Ti GPU-val rendelkező Linux-rendszeren futtattuk az elemzéseket.

A kísérletek során használt modellek leírásai, valamint cikkek referenciái az *intent_prediction/README.md* fájlban kerültek összegyűjtésre.

A kis méretű SZDTROBERTA–7000 modell tesztelésének tanulságait a következő fázisban arra használtuk fel, hogy egy továbbfejlesztett, nagy méretű előtanított modellt állítsunk elő. A kísérletek folytatásához az intézet úgynevezett *dataminer* szerverét jelöltük ki, melyen kialakítottuk a kísérleti környezetet. Ez az erőforrás már rendelkezik GPU-val, ami lehetővé tette számunkra, hogy betanítsunk egy nagyobb méretű alapmodellt, melynek az SZDTROBERTA munkanevet adtuk.

1.2. Az NLU_BENCHMARK magyar nyelvű megfelelőjének elkészítése

További kísérletek elvégzése céljából szükségünk volt egy magyar nyelű osztályozott szövegre, mely tanításra, illetve tesztelésre egyaránt felhasználható. Elhatároztuk, hogy az angol nyelvű 7 osztályos NLU_BENCHMARK tanítómintát automatikusan magyar nyelvre konvertáljuk, és ezzel fejlesztjük tovább a kísérleti rendszerünket, mely ez idáig csupán a mondat ROOT szavának predikciójára volt képes. A konvertált modellnek NLU_BENCHMARK_HU_RAW munkanevet adtuk.

Automatikus fordításhoz a Google *translate* Python API-t, és magát a translate szolgáltatást is felhasználtuk, de az eredmény nem volt kielégítő, mivel karakterkódolási és nyelvhelyességi hibák egyaránt keletkeztek. A keletkezett tanítóminta karakterkódolási problémáit kijavítottuk. Az előállt fájlt ezután manuálisan sorról sorra átnéztük, és az ott található szöveget kijavítottuk. Legjellemzőbb problémaként említhető a fordító hibájából keletkezett sok magyartalan kifejezés. A végleges tanítómintának az NLU_BENCHMARK_HU munkanevet adtuk.

1.3. A saját kísérleti alapmodelljeink és a HIL_ROBERTA alapmodell összehasonlítása

Az SZDTROBERTA–7000 kísérleti alapmodellt és a HIL_ROBERTA alapmodellt behatóbban össze kellett hasonlítanunk. Ennek két célja volt:

1. A HIL_ROBERTA licenc birtokosa a Nyelvtudományi Intézet, ami behatárolja, hogy mire használhatjuk a modellt. E helyett tehát érdemes egy saját alapmodellt használni.
2. A kísérleteink szempontjából az SZDTROBERTA–7000 egy doménspecifikus alapmodellnek tekinthető. Bár a HIL_ROBERTA valószínűleg jobb általános célra, de doménspecifikus feladatokra a saját alapmodell is megfelelő lehet.

A magyar NLU_BENCHMARK_HU_RAW tanítóminta segítségével 4 finomhangolt modell tanítását végeztük el, és az elkészült modelleket manuálisan is kiértékeltük.

A javított magyar NLU_BENCHMARK_HU tanítóminta segítségével a tanítást 3, illetve 6 epochon keresztül újra elvégeztük, először az új SZDTROBERTA modell, majd a HIL_ROBERTA modell finomhangolásával.

1.4. Kapcsolt szándékfeltárás és adatkinyerés

A magyar nyelvű saját alapmodell elkészülte után a fókuszunkat a többszörös szándék-prognosztizáció, illetve a szándékhoz kapcsolódó adatok kinyerésére fordítottuk. Olyan megoldásokat kerestünk, ahol ezen feladatok kapcsolt módon kerülnek megoldásra.

2. Kutatási eredmények összesítése

2.1. Elvégzett kísérletek bemutatása

2.1.1. Alapmodell létrehozása

A magyar nyelvű tanítóminta kiválasztása, előkészítése

A magyar nyelvű kérések feldolgozásához magyar nyelvi modellre, illetve magyar tanítómintára van szükségünk. Az egyik legjelentősebb magyar nyelvű annotált korpuszt, a Szeged Dependency Treebank (SZDT) felhasználását jelöltük ki a kezdeti vizsgálat tárgyaként. Az SZDT beszerzése megtörtént, ebből egy 10 000 mondatos alkorpuszt készítettünk elő a szabad szöveges feldolgozásra. Ezen folyamat során az annotált formátumot annotálatlan szöveges formára alakítottuk, valamint elkészítettünk egy társállományt, mely a mondatok függőségi nyelvtanból ismert, ún. ROOT fogalmát tartalmazza. Ezen két állomány szükséges a szándék-prognosztizációs kísérlet elvégzéséhez. A kísérlet alapja az a feltételezés, hogy a mondat a gyökér szava által definiált szándékot indukálja. Ez a gyökércímke helyettesíti a mondat szándékleíró kódját. Mivel ekkor nem állt rendelkezésre konkrét magyar nyelvű osztályozott tanítóminta-korpusz, ezért a leírt módszerrel kezdtük el a vizsgálatokat.

	utterance	label
9995	Pár perc múlva apa talált egy csavarkulcsot.	talált
9996	Jól összekente vele magát.	összekente
9997	Úgy gondolta, hogy a kilátóhelynél elrejti, vi...	gondolta
9998	Csodálatos volt a kilátás, a hegyeken látszott...	volt
9999	Anya talált egy szarvasbogarat, ezzel ő vezetett.	talált

Részlet az elkészült tanítómintából

A 10 000 mondat ezzel a módszerrel 2924 osztályba volt sorolható. Az előtanított modell a SZDTROBERTA–7000 volt, melyet egy korábbi NLP-kísérletünk során hoztunk létre.

Angol nyelvű tanítóminta kiválasztása, előkészítése

Az úgynevezett NLU-BENCHMARK halmaz alkalmas arra, hogy angol nyelvű szöveg-osztályozó, illetve feladatmegoldó algoritmusokat értékeljünk ki. Összesen 7 osztályt tartalmaz, melyek egyben szándékleíró címkéknek is tekinthetőek.

Utterance	Intent
I would like to hear The Worst Is Yet To Come	*PlayMusic*
Give A History of the Mind a 2 out of 6 points.	*RateBook*

Példák az NLU-BENCHMARK-ból

Minden osztályban 300 mondattöredék (utterance) található:
Class: AddToPlaylist, # utterances: 300
Class: BookRestaurant, # utterances: 300
Class: GetWeather, # utterances: 300
Class: PlayMusic, # utterances: 300
Class: RateBook, # utterances: 300
Class: SearchCreativeWork, # utterances: 300
Class: SearchScreeningEvent, # utterances: 300

Ezen szándékleíró címkéket azonosítókká konvertáltuk:
label to index: {'AddToPlaylist': 0, 'BookRestaurant': 1, 'GetWeather': 2, 'PlayMusic': 3, 'RateBook': 4, 'SearchCreativeWork': 5, 'SearchScreeningEvent': 6}

Előtanított modellként az angol nyelvű ROBERTA-BASE-t használtuk:
config = RobertaConfig.from_pretrained('roberta-base')

Kísérletek a magyar és az angol nyelvű tanítóminták segítségével

NLU-BENCHMARK tanítóminta jellemzői:
FULL Dataset: 2100
TRAIN Dataset: 1680
TEST Dataset: 420
Intent: 7

SZDT–10000 tanítóminta jellemzői:
FULL Dataset: 10 000
TRAIN Dataset: 8000
TEST Dataset: 2000
Intent: 2924

A tanítás az angol nyelvű minta segítségével 3 epochon keresztül zajlott, mely eredménye egy kevéssé rugalmas túltanított modell lett:
Iteration: 1600. Loss: 0.03266046196222305. Accuracy: 100,0%

A magyar nyelvű minta a SZDTROBERTA–7000 használatával, karakter alapú tokenizálással nem hozott értékelhető eredményt:
Iteration: 4600. Loss:7.135164566040039. Accuracy: 10,1%

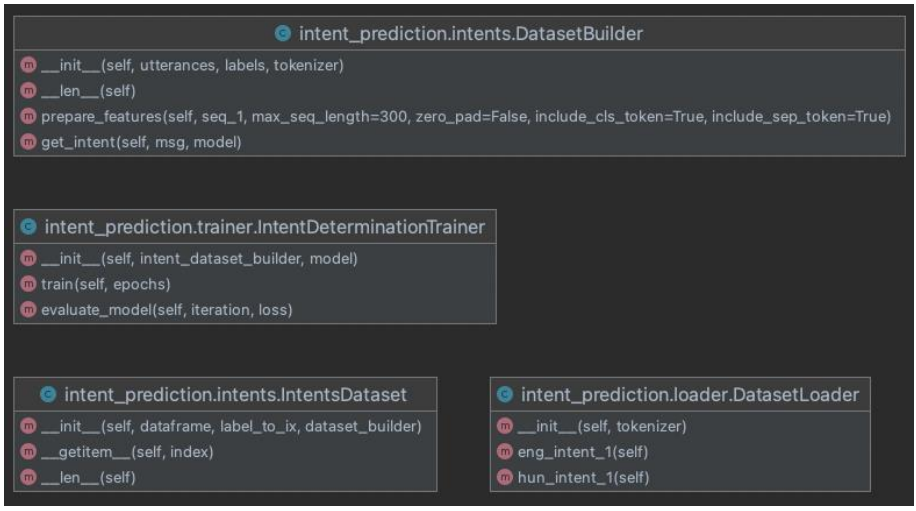
A magyar modell újratanítása teljes szavas tokenizálással

A magyar modellt ismételten betanítottuk teljes szavas tokenizálással. Ennek eredménye lett a **model_2,** mely már értékelhető pontossággal bírt (Accuracy: 36,15%).

Ugyanezen tanítómintával a HIL_ROBERTA modellt is finomhangoltuk, így össze tudjuk mérni az általános alapmodell (HIL_ROBERTA), valamint a doménspecifikus alapmodell (SZDTROBERTA–7000) teljesítményét. A HIL_ROBERTA alapmodell finomhangolását háromféleképpen végezetük el, folyamatosan emelve az epochok számát: 3-ra, 6-ra, majd 12-re.

Python-osztályok

A kísérlet során kifejlesztésre került programrészeket objektumorientált struktúrába szerveztük, mely megkönnyíti a további kísérletek elvégzését, illetve az alrendszer beépítését egy nagyobb struktúrába. A kétnyelvű kísérlet jelenleg a következő osztályokra épül:



Python-osztályok

Az SZDTROBERTA modell elkészítése a teljes SZDT-korpusz felhasználásával

A *dataminer* szerver használata saját, nagy méretű ROBERTA modell elkészítésére: A tanítás utasításait a *roberta.ipynb* notebookfájl tartalmazza. A megfelelő modulok telepítése után egy saját *ByteLevelBPETokenizer* tanítását végeztük el. Ezúttal a teljes SZDT-korpuszt felhasználtuk. Néhány manuális teszteléssel ellenőriztük le a tokenizer működését.

A *RobertaForMaskedLM* Python-modellt előtanítottuk, mely a megadott konfigurációban 83.5 millió paramétert tartalmaz. A definiált transformer tokenizer segítségével létrehoztunk egy data collatort. A collator feladata, hogy a tanítóminta adatait a *Trainer* számára megfelelően kötegelt formátumúra hozza. Végül megadtuk a *Trainer* megfelelő argumentumait és elindítottuk a műveletet.

Mivel GPU-hibák jelentkeztek, elemeztük a beállításokat és kiderítettük, hogy jelen formában a köteg méretét egészen 4-re le kell csökkenteni, hogy a folyamat sikeres legyen:

```
per_device_train_batch_size=4
```

Tanulságként levonható, hogy a folyamatot érdemes először CPU-n kipróbálni, és csak miután hibamentesen elindul a tanítás, akkor aktiválni a GPU-t. Ennek oka, hogy a CPU-val kapcsolatos műveletek sokkal beszédesebb, konkrétabb hibaüzeneteket adnak, amivel jelentősen megkönnyítik az elemzést, hibajavítást.

A kísérletsor célját elértük, egy saját nagy méretű RoBERTa modell létrejött, melynek a SZDTROBERTA munkanevet adtuk.

2.1.2. Az NLU_BENCHMARK magyar nyelvű megfelelőjének elkészítése

Angol nyelvű 7 osztályos tanítóminta konverziója

Feltételeztük, hogy a Google Translate online fordítóalkalmazás (https://translate.google.com) megfelelő választás lesz a tanítóminta konvertálására. Mivel az angol nyelvű minta nem nyers szövegként áll rendelkezésre, ezért az volt a terv, hogy egy Python könyvtár segítségével hajtjuk végre a fordítást. Választásunk a *googletrans* könyvtárra esett:

```
from googletrans import Translator
```

Azonban úgy tűnik a Google Translate backend nem támogatja a nagy tömegű elérést, ezért néhány sikeres request-response után a fordítási folyamat hibával megszakadt.

A tömeges mondatfordítás lehetőségét így elvetettük és helyette exportáltuk mind a 2100 mondattöredéket:

intent_dataset_builder.dataset['utterance'])

Mivel ekkora mennyiségű szöveget nem támogat a Google Translate fájlként sem, ezért 500 soros fájlokra daraboltuk azt, majd az elkészült eredményfájlokat összefűztük. Az elkészült tanítómintát ezután tisztítani és véglegesíteni kellett, hogy használható legyen modell-finomhangolásra. Tömörítés után az állományok bekerültek a git repositoryba a következő néven:

translated_intent.zip
translated_intent_labels.zip

Az adatbetöltő modult frissítettük ezekkel az új állománynevekkel és elkészítettük az új tanító jupyter notebookot:

train_hun_intent_model_2.ipynb

Az elkészült tanítómintának az NLU_BENCHMARK_HU_RAW munkanevet adtuk.

2.1.3. A saját kísérleti alapmodelljeink és a HIL_ROBERTA alapmodell összehasonlítása

Az SZDTROBERTA–7000 és a HIL_ROBERTA alapmodelleket több tanítóminta finomhangolása segítségével is összehasonlítottuk. Az SZDT–10000 minta volt az első kísérletünk tárgya. Többféle tokenizációs eljárást, illetve többféle epoch beállítást is teszteltünk, a cél egyetlen szándék, a mondat ROOT kifejezésének meghatározása volt.

A két alapmodell összehasonlítását az automatikusan magyar nyelvre fordított NLU_BENCHMARK_HU_RAW tanítóminta felhasználásával folytattuk. Itt már képesek voltunk a definiált 7 szándék detektálási sikerességének mérésére, illetve összehasonlítására. Továbbá néhány saját mondatot is megadtunk, melyek segítségével manuális összehasonlítást is végeztünk.

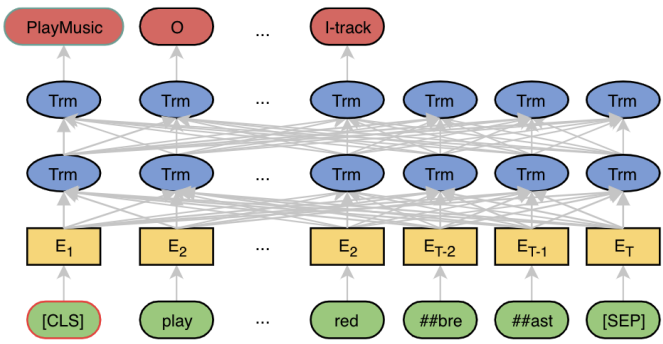
Az NLU_BENCHMARK_HU_RAW tanítóminta manuális javításával előállt NLU_BENCHMARK_HU tanítóminta segítségével, valamint a nagy méretű SZDTROBERTA alapmodellünk felhasználásával 4 modell finomhangolását végeztük el, és az elkészült modelleket manuálisan is kiértékeltük, valamint összevetettük a HIL_ROBERTA alapmodell finomhangolásából származó eredményekkel. Konklúzióként megállapítottuk, hogy a saját alapmodellünk fölveszi a versenyt a Nyelvtudományi Intézet modelljével, így a továbbiakban a saját alapmodellünket fogjuk a kísérletekre használni.

2.1.4. Kapcsolt szándékfeltárás és adatkinyerés

Az irodalom alapján Gangadharaiah et al. a szándékdetektálási feladatot mint többszörös címkézés (multi-label classification) oldja meg. Xu et al. (2013) munkájára építenek, akik a feladathoz log-linear módszert használnak. Kiegészítésképpen Gangadharaiah et al. slot labeling műveletet is végeznek. Példaként tekintsük a következőket: ugyanazon domén két különböző szándékot is magában foglal: „BookCab" és „BookHotel". A „BookCab" szándékhoz a következő három label kapcsolható: „City", „Time" és „Loc". A „BookHotel" szándékhoz a „City", „CheckinDate" és a „Duration". A következő szövegben: „book a cab from the airport in Seattle and find me a hotel to stay" a felhasználó hotelt és taxit is foglalni kíván, vagyis mindkét szándék aktív. A megjelenő „City" label: „Seattle" mindkét szándékhoz kapcsolódik. A feladatok megoldásához BiLSTM neurális hálózatot használnak encoder-decoder összeállításban. Szóbeágyazásként egy külön réteget használnak előtanított beágyazások helyett.

Chen et al. előtanított BERT-modell felhasználásával oldja meg a szándékprognosztizációs feladatot. Javaslatuk szerint egy speciális klasszifikációs tokent ([CLS]) illesztenek a szószekvencia elejére, valamint egy elválasztó ([SEP]) tokent a szekvencia végére. A [CLS] token hidden state-jét használva prognosztizálható a szándék. A többi token hidden state-jének felhasználásával állapítható meg a slot label. Mivel WordPiece tokenizálót használnak, ezért a tokenek első sub-Tokenjének hidden state-jét használják a teljes tokenek helyett. Javasolt megoldásukkal kimagasló eredményeket produkálnak.

A transformer modell bemeneti oldalán nem csupán a mondatok tokenizált formája jelenik meg, hozzáfűzésre kerül egy speciális token a mondat elejére [CLS] valamint a végére [SEP].



A Transformer-modell tokenkezelése

Számunkra a [CLS] token a legizgalmasabb. Az ábra bemutatja, hogy a transformer modell outputján a [CLS] token a szándékleíró label értékét veszi fel. Igaz, hogy a [CLS] token értéke a bemeneti oldalon mindig ugyanaz, azonban a transformer modell környezetfüggő (contextualized), így a kimenet a szövegkörnyezettől függően más és más lehet. Finomhangolás során tehát ezzel a módszerrel vagyunk képesek a mondat klasszifikációjára.

Mint látszik, a többi mondatalkotó token (pl.: szó, írásjel) adatleíró címkévé transzformálódik. Ez a két feladat (intent detection & slot filling) tehát kapcsolt módon oldható meg a transformer modell segítségével.

Az elemzések következő fázisában elkezdtük a CLS-token használatának beépítését a kísérleti rendszerünkbe. A *DatasetBuilder* osztály, mely előkészíti a nyers szövegből az inputvektort, tartalmaz egy *prepare_features()* függvényt, mely paramétertől függően hozzáragasztja az inputvektorhoz a CLS- és SEP-tokenek reprezentációját. Azért, hogy ezen függvény működése érthetőbb legyen, implementáltunk hozzá egy egységtesztet: *test_data_loader()*.

Mivel a kísérleti rendszerünk modellstruktúrája egyszeres szándékfeltárásra készült, ezért ebben a formájában nem alkalmas arra, hogy kapcsolt többszörös szándékfeltárást és szekvenciális címkézést végezzünk vele. Jó alapot biztosít azonban a továbbfejlesztésre. A GitHubon elérhető JointBERT algoritmus forráskódelemzése megmutatta, hogy a kapcsolt végrehajtás megoldható BERT, DistilBERT és alBERT modellek esetén, ezért jó esély van arra, hogy RoBERTa modellnél is alkalmazható. A kapcsolt modell lényegét tekintve a kimeneten mind az *intent*, mind a *slot* megjelenik, melyhez párosul az előző réteg hidden_state-je is. Ezt a megoldást kell beépítenünk a RoBERTa modellbe is.

```
outputs = ((intent_logits, slot_logits),) + outputs[2:]  # add hidden states and attention if they are here

outputs = (total_loss,) + outputs

return outputs  # (loss), logits, (hidden_states), (attentions) # Logits is a tuple of intent and slot logits
```

A tanítóminta módosítására is szükségünk van, mivel a mondattöredékek, illetve szándékkategóriák meghatározása mellett a szekvenciális címkéket is felhasználunk. A következő példák az ún. ATIS (Airline Travel Information System) adathalmazból származnak. Jól látható, hogy a mondattöredék, a hozzá tartozó kategória, valamint a szekvenciális címkék listája 3 fájlban kerül tárolásra. Összekötésük a sorszám alapján történik.

```
18  i'm looking for a flight from oakland to denver with a stopover in dallas fort worth
19  what's restriction ap68
20  what types of ground transportation are available in philadelphia
21  what does the abbreviation co mean
22  a first class flight on american to san francisco on the coming tuesday
```

```
18  atis_flight
19  atis_restriction
20  atis_ground_service
21  atis_abbreviation
22  atis_flight
```

```
18  O O O O O O B-fromloc.city_name O B-toloc.city_name O O O O B-stoploc.city_name I-stoplo
19  O O B-restriction_code
20  O O O O O O O O B-city_name
21  O O O O B-airline_code O
22  O B-class_type I-class_type O O B-airline_name O B-toloc.city_name I-toloc.city_name O O
```

Ahhoz, hogy a szükséges tanítómintát elkészíthessük, szükségünk van egy annotációkészítő alkalmazásra. A finomhangolás során a transzformer modell bemeneti oldalán szakterületspecifikus mondatok jelennek meg (text), melyhez egy annotációszekvenciát párosítunk (labels). A szöveg ellátása annotációval egy monoton feladat, melyre számos megoldás létezik napjainkban. Ezen megoldások áttekintését elvégeztük. Az elemzés során megpróbáltuk azokat a kategóriákat áttekinteni, melyek számunkra leginkább relevánsak voltak.

A fent említett kapcsolt megoldás még nem teljesen fedi le a kutatási tervben ismertetett kívánalmakat, ezért tanulmányoztuk, hogyan tudnánk a többszörös kategorizálást, vagyis a többszörös szándékprognosztizálást megvalósítani. Ebben az esetben a kategória egy vektorként definiálható, mely azokon a pozíciókon veszi fel az 1 értéket, ahol a pozíciónak megfelelő kategória a mondatra érvényes, a többi pozíción 0. Jó példa erre, az angol nyelvű ToxicComments adathalmaz.

```
train_df = pd.read_csv('data/train.csv')
train_df.head()
```

Out[2]:

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
	32d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
	d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
	7ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
	1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
	54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

2.2. Kiértékelések eredményeinek bemutatása

2.2.1. SZDT–10000 tanítómintán végzett kísérletek

Az SZDTROBERTA–7000 és a HIL_ROBERTA alapmodellek segítségével végzett egyszeres szándékmeghatárolási kísérletek eredménye azt mutatta, hogy a Nyelvtudományi Intézet modellje jelentősen pontosabb, mint a saját kis méretű modellünk.

Model ID	Tokenizer and model	Tokenize	Epoch	Accuracy
model_1	SZDTROBERTA–7000	to characters	3	10,05%
model_2	SZDTROBERTA–7000	to words	3	36,15%
model_3	HIL_ROBERTA	to words and word parts	3	33,75%
model_4	HIL_ROBERTA	to words and word parts	6	47,05%
model_5	HIL_ROBERTA	to words and word parts	12	74,10%

A kiértékelés összefoglalása: SZDT–10000 tanítóminta, egyetlen szándék

A modellek manuális tesztelése is megtörtént néhány, a tanítómintában nem szereplő mondattal. Ebben az esetben is a model_5 lett a legeredményesebb.

a) Kirándulni mentünk.
b) Nagyon érdekes és szép volt ez a dombok övezte hely.
c) A boltba vittünk néhány almát.
d) Mit gondolsz a világról?
e) Sajnálom, hogy így elromlott az autó!

	model_1	model_2	model_3	model_4	model_5
a)	elindultunk	mentünk	mentünk	mentünk	mentünk
b)	elindultunk	volt	volt	volt	volt
c)	elindultunk	van	volt	van	vittünk
d)	elindultunk	eljött	van	van	gondolok
e)	elindultunk	van	van	van	van

Az egyes modellek által megállapított szándék

2.2.2. NLU_BENCHMARK tanítómintán végzett kísérletek

Az angol nyelvű modell manuális kiértékelése néhány saját mondat segítségével szintén megtörtént. Az eredmény a várakozásoknak megfelelő lett.

Utterance	Detected intent
Is it sunny today?	*GetWeather*
I would like to hear a nice song!	*PlayMusic*
Is it sunny today? I would like to hear a nice song!	*BookRestaurant*

Angol mondatok és a megállapított szándék

2.2.3. Magyar nyelvre automatikusan fordított NLU_BENCHMARK_HU_RAW tanítómintán végzett kísérletek

A tanítást 3, illetve 6 epochon keresztül végeztük, először az SZDTROBERTA–7000 alapmodell, majd a HIL_ROBERTA alapmodell finomhangolásával. Célunk az volt, hogy a két modell összehasonlítható legyen a szándékdetektálási feladatok megoldása szempontjából.

model_1:
SZDTROBERTA–7000 tokenizer and model, epoch 3, Accuracy: 91,9%
model_2:
HIL_ROBERTA tokenizer and model, epoch 3, Accuracy: 94,52%

model_3:
SZDTROBERTA–7000 tokenizer and model, epoch 6, Accuracy: 97,14%
model_4:
HIL_ROBERTA tokenizer and model, epoch 6, Accuracy: 97,61%

A modellek manuális kiértékelése

AddToPlaylist:
1. Tedd a listámra a *Sose halunk meg* című számot.
2. Add a kedvencekhez Michael Jacksontól a Billie Jeant.

BookRestaurant:
3. Ma estére foglalj asztalt az ablak mellé.
4. Ma ebédelni szeretnék valahol.

GetWeather:

5. Milyen idő lesz holnap reggel?
6. Elmehetek vajon sétálni az este?

PlayMusic:

7. Játszd le a kedvenc számomat!
8. Jó lenne hallani valami zenét.

RateBook:

9. A Harry Potter-könyveknek adj egy kilences értékelést.
10. A reggel olvasott könyvem nagyon gyenge.

SearchCreativeWork:

11. Hol található Michelangelo leghíresebb szobra, a David?
12. Keresd meg nekem a *Nyomorultak* című művet!

SearchScreeningEvent:

13. Milyen filmeket adnak a mozik?
14. Hol láthatok egy jó vígjátékot?

	model_1	**model_2**	**model_3**	**model_4**
1	AddToPlaylist	AddToPlaylist	AddToPlaylist	AddToPlaylist
2	SearchScreeningEvent	GetWeather	SearchScreeningEvent	SearchCreativeWork
3	PlayMusic	BookRestaurant	BookRestaurant	BookRestaurant
4	PlayMusic	GetWeather	GetWeather	BookRestaurant
5	GetWeather	GetWeather	GetWeather	GetWeather
6	SearchScreeningEvent	GetWeather	GetWeather	SearchCreativeWork
7	PlayMusic	PlayMusic	PlayMusic	PlayMusic
8	PlayMusic	PlayMusic	PlayMusic	PlayMusic
9	RateBook	RateBook	BookRestaurant	SearchCreativeWork
10	RateBook	BookRestaurant	SearchScreeningEvent	SearchCreativeWork
11	SearchScreeningEvent	GetWeather	BookRestaurant	SearchScreeningEvent
12	SearchCreativeWork	SearchCreativeWork	SearchCreativeWork	SearchCreativeWork
13	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent
14	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent

2.2.4. Az SZDTROBERTA modell tanításjellemzői, statisztikák:

***** Running training *****
Num examples = 81 967
Num Epochs = 1
Instantaneous batch size per device = 4
Total train batch size (w. parallel, distributed & accumulation) = 4
Gradient Accumulation steps = 1
Total optimization steps = 20 492

CPU times: user 32 min, sys: 6.74 s, total: 32 min 7 s
Wall time: 32 min 2 s

TrainOutput(global_step=20492, training_loss=7.5551160447140955, metrics={
'train_runtime': 1922.061,
'train_samples_per_second': 42.645,
'train_steps_per_second': 10.661,
'total_flos': 825760901437056.0,
'train_loss': 7.5551160447140955,
'epoch': 1.0})

2.2.5. Magyar nyelvre fordított tanítóminta manuális javítása után előállt NLU_BENCHMARK_HU tanítómintán végzett kísérletek

Az NLU_BENCHMARK_HU tanítómintával az SZDTROBERTA alapmodellt finomhangoltuk.
A harmadik epoch végére elért Loss, illetve Accuracy:
Loss: 0.0303784366697073. Accuracy: 97,14%

A hatodik epoch végére elért Loss, illetve Accuracy:
Loss: 0.0008110094931907952. Accuracy: 98,01%

A HIL_ROBERTA alapmodellen is elvégeztük ugyanezen műveletsort.
A harmadik epoch végére elért Loss, illetve Accuracy:
Loss: 0.04047880321741104. Accuracy: 96,66666666666667%

A hatodik epoch végére elért Loss, illetve Accuracy:
Loss: 0.00409882515668869. Accuracy: 99,76190476190476%

A modellek kódjai, és eredményei:

model_1:
SZDTROBERTA tokenizer and model, epoch: 3, Accuracy: 97,1%

model_2:
SZDTROBERTA tokenizer and model, epoch: 6, Accuracy: 98,1%

model_3:
HIL_ROBERTA tokenizer and model, epoch: 3, Accuracy: 96,7%

model_4:
HIL_ROBERTA tokenizer and model, epoch: 6, Accuracy: 99,8%

A már korábban megadott 14 saját mondattal elvégeztük a manuális tesztelést mind a négy modell felhasználásával. A hibás predikciók száma csökkent.

	model_1	**model_2**	**model_3**	**model_4**
1	AddToPlaylist	AddToPlaylist	AddToPlaylist	AddToPlaylist
2	SearchScreeningEvent	SearchScreeningEvent	GetWeather	SearchScreeningEvent
3	BookRestaurant	BookRestaurant	BookRestaurant	BookRestaurant
4	BookRestaurant	BookRestaurant	GetWeather	GetWeather
5	GetWeather	GetWeather	GetWeather	GetWeather
6	GetWeather	RateBook	GetWeather	GetWeather
7	PlayMusic	PlayMusic	PlayMusic	PlayMusic
8	PlayMusic	PlayMusic	PlayMusic	PlayMusic
9	SearchCreativeWork	RateBook	RateBook	RateBook
10	SearchCreativeWork	RateBook	RateBook	RateBook
11	SearchScreeningEvent	SearchScreeningEvent	GetWeather	SearchScreeningEvent
12	SearchCreativeWork	SearchCreativeWork	SearchCreativeWork	SearchCreativeWork
13	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent	SearchScreeningEvent
14	SearchScreeningEvent	SearchScreeningEvent	GetWeather	SearchScreeningEvent

3. Technológiai folyamatok bemutatása

Technológiai folyamat szöveges ismertetése

Magyar SZDT–10000 tanítóminta feldolgozásának finomhangolási konfigurációja. Alapmodell: SZDTROBERTA–7000


```
{
 "architectures": [
   "RobertaForMaskedLM"
 ],
 "attention_probs_dropout_prob": 0.1,
 "bos_token_id": 0,
 "classifier_dropout": null,
 "eos_token_id": 2,
 "finetuning_task": null,
 "gradient_checkpointing": false,
 "hidden_act": "gelu",
 "hidden_dropout_prob": 0.1,
 "hidden_size": 768,
 "initializer_range": 0.02,
 "intermediate_size": 3072,
 "layer_norm_eps": 1e-12,
 "max_position_embeddings": 514,
 "model_type": "roberta",
 "num_attention_heads": 12,
 "num_hidden_layers": 6,
 "num_labels": 2924,
 "output_attentions": false,
 "output_hidden_states": false,
 "pad_token_id": 1,
 "position_embedding_type": "absolute",
 "pruned_heads": {},
 "torch_dtype": "float32",
 "torchscript": false,
 "transformers_version": "4.11.0.dev0",
 "type_vocab_size": 1,
 "use_cache": true,
 "vocab_size": 52000
}
```

Angol nyelvű NLU-BENCHMARK tanítóminta finomhangolási konfigurációja. Alapmodell: ROBERTA_BASE


```
{
 "architectures": [
   "RobertaForMaskedLM"
 ],
 "attention_probs_dropout_prob": 0.1,
 "bos_token_id": 0,
 "eos_token_id": 2,
 "finetuning_task": null,
 "hidden_act": "gelu",
 "hidden_dropout_prob": 0.1,
 "hidden_size": 768,
 "initializer_range": 0.02,
 "intermediate_size": 3072,
 "layer_norm_eps": 1e-05,
 "max_position_embeddings": 514,
 "model_type": "roberta",
 "num_attention_heads": 12,
 "num_hidden_layers": 12,
 "num_labels": 7,
 "output_attentions": false,
 "output_hidden_states": false,
 "pad_token_id": 1,
 "pruned_heads": {},
 "torchscript": false,
 "type_vocab_size": 1,
 "vocab_size": 50265
}
```

Magyar nyelvre automatikusan fordított NLU-BENCHMARK_HU_1 tanítóminta finomhangolási konfigurációja.
Alapmodell: SZDTROBERTA–7000


```
{
"architectures": [
"RobertaForMaskedLM"
],
"attention_probs_dropout_prob": 0.1,
"bos_token_id": 0,
"classifier_dropout": null,
"eos_token_id": 2,
"finetuning_task": null,
"gradient_checkpointing": false,
"hidden_act": "gelu",
"hidden_dropout_prob": 0.1,
"hidden_size": 768,
"initializer_range": 0.02,
"intermediate_size": 3072,
"layer_norm_eps": 1e-12,
"max_position_embeddings": 514,
"model_type": "roberta",
"num_attention_heads": 12,
"num_hidden_layers": 6,
"num_labels": 7,
"output_attentions": false,
"output_hidden_states": false,
"pad_token_id": 1,
"position_embedding_type": "absolute",
"pruned_heads": {},
"torch_dtype": "float32",
"torchscript": false,
"transformers_version": "4.11.0.dev0",
"type_vocab_size": 1,
"use_cache": true,
"vocab_size": 52000
}
```

Magyar nyelvre automatikusan fordított NLU-BENCHMARK_HU_1 tanítóminta finomhangolási konfigurációja alapmodell: HIL_ROBERTA


```
{
"architectures": [
"RobertaForMaskedLM"
],
"attention_probs_dropout_prob": 0.1,
"bos_token_id": 0,
"eos_token_id": 2,
"finetuning_task": null,
"gradient_checkpointing": false,
"hidden_act": "gelu",
"hidden_dropout_prob": 0.1,
"hidden_size": 768,
"initializer_range": 0.02,
"intermediate_size": 3072,
"layer_norm_eps": 1e-12,
"max_position_embeddings": 514,
"model_type": "roberta",
"num_attention_heads": 12,
"num_hidden_layers": 6,
"num_labels": 7,
"output_attentions": false,
"output_hidden_states": false,
"pad_token_id": 1,
"pruned_heads": {},
"torchscript": false,
"type_vocab_size": 1,
"vocab_size": 30000
}
```

Az SZDTROBERTA alapmodell konfigurációja


```
{
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "eos_token_id": 2,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 514,
  "model_type": "roberta",
  "num_attention_heads": 12,
  "num_hidden_layers": 6,
  "pad_token_id": 1,
  "position_embedding_type": "absolute",
  "transformers_version": "4.14.0.dev0",
  "type_vocab_size": 1,
  "use_cache": true,
  "vocab_size": 52000
}
```

A javított magyar NLU-BENCHMARK_HU tanítóminta finomhangolási konfigurációja. Alapmodell: SZDTROBERTA


```
{
"architectures": [
"RobertaForMaskedLM"
],
"attention_probs_dropout_prob": 0.1,
"bos_token_id": 0,
"classifier_dropout": null,
"eos_token_id": 2,
"finetuning_task": null,
"hidden_act": "gelu",
"hidden_dropout_prob": 0.1,
"hidden_size": 768,
"initializer_range": 0.02,
"intermediate_size": 3072,
"layer_norm_eps": 1e-12,
"max_position_embeddings": 514,
"model_type": "roberta",
"num_attention_heads": 12,
"num_hidden_layers": 6,
"num_labels": 7,
"output_attentions": false,
"output_hidden_states": false,
"pad_token_id": 1,
"position_embedding_type": "absolute",
"pruned_heads": {},
"torch_dtype": "float32",
"torchscript": false,
"transformers_version": "4.14.0.dev0",
"type_vocab_size": 1,
"use_cache": true,
"vocab_size": 52000
}
```

A javított magyar NLU-BENCHMARK_HU tanítóminta finomhangolási konfigurációja. Alapmodell: HIL_ROBERTA


```
{
"architectures": [
"RobertaForMaskedLM"
],
"attention_probs_dropout_prob": 0.1,
"bos_token_id": 0,
"eos_token_id": 2,
"finetuning_task": null,
"gradient_checkpointing": false,
"hidden_act": "gelu",
"hidden_dropout_prob": 0.1,
"hidden_size": 768,
"initializer_range": 0.02,
"intermediate_size": 3072,
"layer_norm_eps": 1e-12,
"max_position_embeddings": 514,
"model_type": "roberta",
"num_attention_heads": 12,
"num_hidden_layers": 6,
"num_labels": 7,
"output_attentions": false,
"output_hidden_states": false,
"pad_token_id": 1,
"pruned_heads": {},
"torchscript": false,
"type_vocab_size": 1,
"vocab_size": 30000
}
```


4. Összegzés

Létrehoztuk a magyar nyelvű NLU_BENCHMARK_HU tanítómintát. Az automatikus fordítás után keletkezett tanítóminta karakterkódolási problémáit kijavítottuk. Az előállt tanítófájl nyelvhelyességét ezután manuálisan sorról sorra kijavítottuk. Így a fájl magyar nyelvű szövegosztályozó, szándékmeghatározó algoritmusok tesztelésére vált alkalmassá.

Elkészítettük az előtanított SZDTROBERTA alapmodellt. Mind az SZDTROBERTA, mind a Nyelvtudományi Intézet HIL_ROBERTA modelljén kísérleteket futtattunk, és összevetettük az eredményeket.

Az eddigi tapasztalatok alapján szükséges megállapítani hogyan kell a meglévő tanítómintát úgy módosítani, hogy a többszörös szándékfeltárás finomhangolása elvégezhető legyen. A finomhangoláshoz szükségünk van egy annotációkészítő alkalmazásra. Kiválasztottunk és elemeztünk néhány ilyen eszközt.

Elemeztük, hogy lehetséges-e a meglévő modellstruktúrával kapcsolt többszörös szándékfeltárást, illetve szekvenciális címkézést végeznünk. A tapasztalat szerint tovább kell fejlesztenünk a jelenlegi megoldást, ennek mikéntjét meghatároztuk, illetve definiáltuk milyen tanítóhalmaz kiegészítésre van szükségünk.