Integracija chatbota u Facebook Messenger

Bosanac, Ljubica

Master's thesis / Diplomski rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:289667

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij

INTEGRACIJA CHATBOTA U FACEBOOK MESSENGER

Diplomski rad

Ljubica Bosanac

Osijek, 2018.

SADRŽAJ

1. UVOD 1

1.1. Zadatak diplomskog rada 1

2. PRIMJENA STROJNOG UČENJA ZA CHATBOT SUSTAVE 3

2.1. Umjetna inteligencija 3

2.1.1. Povijest 3

2.1.2. Strojno učenje 8

2.1.3. Duboko učenje 10

2.2. Obrada prirodnog jezika 12

2.2.1. Pristupi 13

2.2.2. Razumijevanje prirodnog jezika 16

2.2.3. Generiranje prirodnog jezika 17

2.3. NLP servisi 19

2.3.1. Dialogflow (Api.ai) 19

2.3.2. Wit.ai 19

2.3.3. IBM Watson 20

3. CHATBOT SUSTAVI 21

3.1. Sustavi temeljeni na pravilima 21

3.2. Sustavi temeljeni na korpusu 23

3.2.1. IR sustavi 23

3.2.2. Sustavi slijeda sekvenci 24

3.3. Trenutno stanje chatbot industrije 25

3.4. Integracija NLP servisa u chatbot 29

3.4.1. Pretprocesiranje podataka 29

3.4.2. Izrada modela 30

3.4.3. Treniranje modela 30

3.4.4. Testiranje ... 31
4. FERITTO CHATBOT ... 32
4.1. Facebook Messenger ... 32
4.2. Izrada chatbota ... 32
4.3. Integracija u Facebook Messenger ... 36
4.4. Funkcionalnost i testiranje ... 38
5. ZAKLJUČAK ... 43
LITERATURA ... 44
POPIS KRATICA ... 47
SAŽETAK ... 48
ABSTRACT ... 49
ŽIVOTOPIS ... 50
PRILOZI ... 51
Prilog 1 – sadržaj listen.rb datoteke ... 51
Prilog 2 – tablica mogućih odgovora ... 60

1. UVOD

Strojevi s ljudskim sposobnostima oduvijek su intrigirali ljude u znanstvenim i umjetničkim krugovima. Opisani su u književnosti, filmu, slikarstvu, a korijene vuku iz inženjerstva, statistike, filozofije, logike, psihologije i drugih znanstvenih i humanističkih grana.

"On se ponaša kao da ima stvarne emocije. Hm, naravno, programiran je na taj način da nama bude lakše razgovarati s njim. Ali ima li stvarne emocije ili ne, nešto je na što vjerujem nitko ne može istinski odgovoriti."[1]

Napredak tehnologije utjecao je na način na koji su psiholozi počeli promatrati ljudski um, a razmišljanja o prenošenju inteligentnog ponašanja na računala pojavila su se samim razvojem računala. Općenito, umjetna inteligencija znanost je koja omogućuje računalima izvršavanje ljudskih djelatnosti. Obrada prirodnog jezika podrazumijeva metode koje umjetnoj inteligenciji omogućuju razumijevanje ljudskog jezika.

1.1. Zadatak diplomskog rada

Tema diplomskog rada je istražiti i opisati metode integracije *chatbota* u Facebook Messenger, istražiti i opisati trenutno *state of the art* metode za obrade prirodnog jezika, istražiti i opisati trenutno dostupne servise za obradu prirodnog jezika koje je moguće integrirati u *chatbot*. U praktičnom dijelu potrebno je izraditi jednostavan *chatbot* i integrirati ga u Facebook Messenger. Kao što je i navedeno, tema diplomskog rada tiče se integracije *chatbota* u Facebook Messenger, a sadržajno će pokriti teme umjetne inteligencije, strojnog i dubokog učenja, *chatbota,* obrade prirodnog jezika i njihovih servisa koje je moguće integrirati u *chatbot,* povijest opisanih područja i dr.

Prvo poglavlje tiče se primjene strojnog učenja za *chatbot* sustave. Objasnit će se umjetna inteligencija kao znanost koja omogućuje računalima izvršavanje ljudskih djelatnosti, njena povijest te pristupi koji uključuju strojno i duboko učenje, zatim obrada prirodnog jezika koja se tiče programiranja računala da obrađuju i analiziraju velike količine podataka u kojima je sadržan ljudski jezik i na samom kraju poglavlja predstavit će se najpoznatiji servisi za obradu prirodnog jezika. U drugom poglavlju objasnit će se *chatbot* sustavi te dati teorijska pozadina nužna za razumijevanje njihovog rada. Opisat će se podjela *chatbot* sustava na sustave

[1] Citat iz filma 2001.: Odiseja u svemiru, slavnog britansko-američkog znanstveno-fantastičnog filma iz 1968. godine snimljenom prema kratkoj priči *The Sentinel* Arthur C. Clarkea. Hal 9000 je računalo kome je povjerena kontrola leta, a u navedenom citatu doktor David Bowman priča upravo o njemu

temeljene na pravilima i sustave temeljene na korpusu, dati pregled trenutnog stanja *chatbot* industrije te opisati načini integracije servisa za obradu prirodnog jezika u *chatbot* sustave, tj. kako dati ljudsku komponentu *chatbot* sustavu. U praktičnom dijelu, ujedno i posljednjem poglavlju, izradit će se *chatbot* te integrirati u Facebook Messenger, a koraci će biti detaljno objašnjeni.

2. PRIMJENA STROJNOG UČENJA ZA CHATBOT SUSTAVE

Za razumijevanja rada *chatbot* sustava potrebno je poznavanje umjetne inteligencije, strojnog učenja, dubokog učenja, obrade prirodnog jezika te razumijevanja i generiranja prirodnog jezika. Prema [1] umjetna inteligencija znanost je koja omogućuje računalima izvršavanje ljudskih djelatnosti, a počeci sežu u sredinu prošlog stoljeća. Već od samih razvoja računala, ljudi su im htjeli dati ljudsku komponentu. Strojno učenje pristup je koji omogućuje umjetnoj inteligenciji učenje iz iskustva kako bi pronašla uzorke u podacima i znala kako se ponašati u novim situacijama. Strojno učenje može biti nadzirano, nenadzirano i podržano [2]. U nadziranom postoje dvije vrste modela: regresija i klasifikacija. Duboko učenje tehnika je implementiranja strojnog učenja kroz strukture koje se nazivaju neuronske mreže.

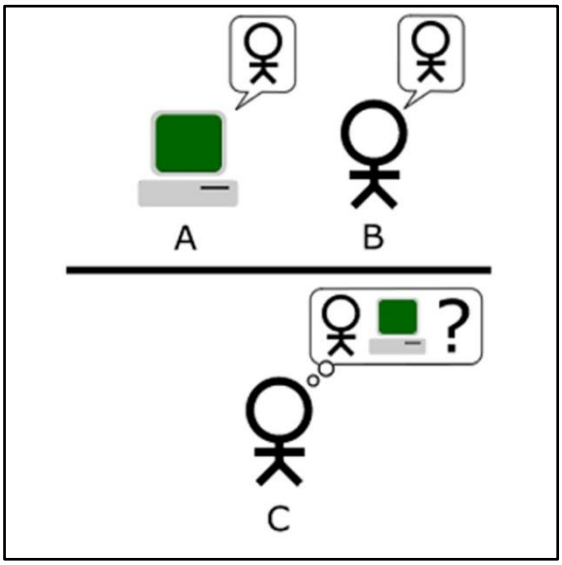
2.1. Umjetna inteligencija

Ljudska težnja davanju ljudske komponente računalima dovela je do razvoja nove znanstvene discipline. Prema [1] umjetna inteligencija znanost je koja omogućuje računalima izvršavanje ljudskih djelatnosti. Neki zadaci vezani za ponašanja su jednostavni za ljude, ali teški za računala. Za rješavanje takvih problema koriste se sustavi koje razvija umjetna inteligencija. Postoje dva pristupa umjetnoj inteligenciji: tradicionalni i alternativni. Tradicionalni pristup znanje predstavlja simbolički, a zaključivanje se opisuje kao manipulacija tih simbola. Kod alternativnog pristupa oponašaju se modeli koji se nalaze u prirodi.

2.1.1. Povijest

Ljudi su računalima od samih početaka njihovog razvoja htjeli dati ljudsku komponentu, a ozbiljnije zanimanje za umjetnu inteligenciju započinje sredinom prošlog stoljeća. Prema [1] 1943. godine McCulloch, i Pitts stvaraju model umjetnog neurona, a 1949. Hebb stvara pravilo za modificiranje veze između dva neurona. 1951. Minsky i Edmons razvijaju prvu neuronsku mrežu koja se sastojala od 40 neurona (vakumskih cijevi). Pojava umjetne inteligencije službeno u povijesti datira 1956. godine kada je na Dartmouth College-u predstavljena prvi put konferencija o umjetnoj inteligenciji, Dartmutska konferencija. Na konferenciji nisu ostvareni neki novi rezultati i rješenja, ali je ustanovljeno novo područje, umjetna inteligencija, koje se razlikovalo od prethodnih istraživanja i teorija vezanih za slična pitanja. 1950. godine Alan Turing je člankom „*Computing Machinery and Intelligence*“ predložio test kojim se testira inteligencija strojeva na način da stroj treba imati određene sposobnosti da „prođe“ test. Neke od sposobnosti su prirodni jezik, odnosno komunikacija na određenom jeziku, reprezentacija znanja, automatsko zaključivanje i sposobnost učenja. Turingov test provodi se na način da

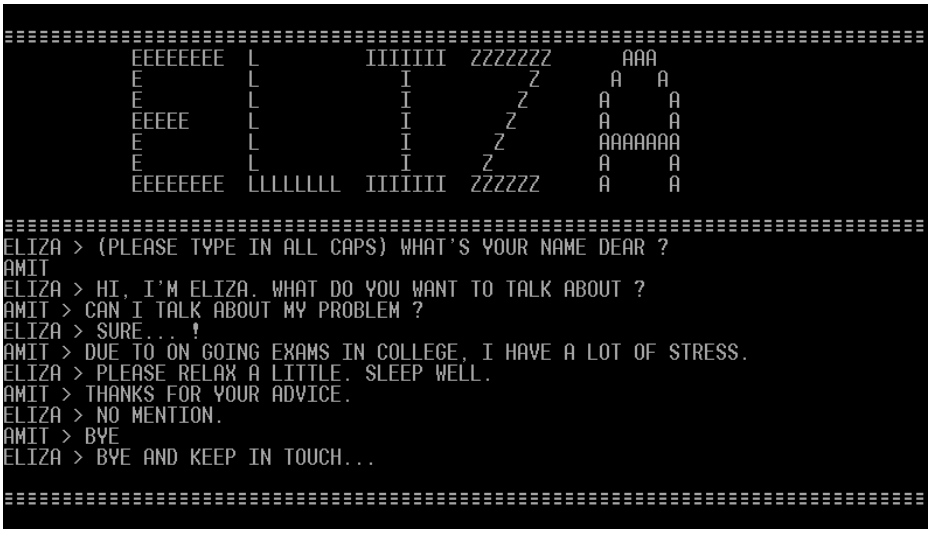
ispitivač vodi dva razgovora: jedan u kojem je sugovornik računalo, a drugi u kojem je sugovornik čovjek, a oba putem računala. Grafički prikaz Turingovog testa prikazan je na Slici 2.1. Tijekom razgovora, ispitivač može postavljati bilo kakva pitanja i voditi razgovor u bilo kom smjeru. Cilj je otkriti u kojoj situaciji razgovara s čovjekom, a u kojoj s računalom. Ako ispitivač pogriješi u više od 30% slučajeva, tj. računalo zamijeni čovjekom, za računalo se smatra da je „prošlo“. 7. lipnja 2014. godine *chatbot* imena Eugene Goostman koji simulira 13-godišnjeg Ukrajinca uvjerio je 33% sudaca Kraljevskog društva Londona da je čovjek te postoji kontroverza oko njegovog „prolaska“ Turingovog testa. Eugene Goostman nije računalo, već računalni program koji se poigrao pravilima i unaprijed organizatorima predstavio priču o sebi, što se nije smjelo učiniti. Tako su organizatori njegove neobične odgovore pripisivali tome što je on zapravo tinejdžer, a ne računalo ili računalni program.



Slika 2.1. *Grafički prikaz Turingovog testa*

1957. godine John McCarthy razvio je LISP (engl. *List Processing Language*), funkcionalni programski jezik za umjetnu inteligenciju. Jezik omogućuje stvaranje fleksibilnih programa koji predstavljaju osnovne operacije s listama struktura.

ELIZA je program za obradu prirodnog jezika kojeg je razvio Joseph Weizenbaum na MIT-evom laboratoriju za umjetnu inteligenciju u razdoblju od 1964. do 1966. godine. U razgovoru oponaša psihoterapeuta, tj. vodi razgovor po uzoru na C. Rogersa, psihoterapeuta koji inače provodi „terapiju usmjerenu na klijenta“. Razgovor se simulira koristeći metodologije supstitucije i slaganja uzoraka koje korisniku daju iluziju razumijevanja, ali program nije imao implementiranu sposobnost tumačenja događaja. Na Slici 2.2. vidljiv je primjer jednog takvog razgovora.



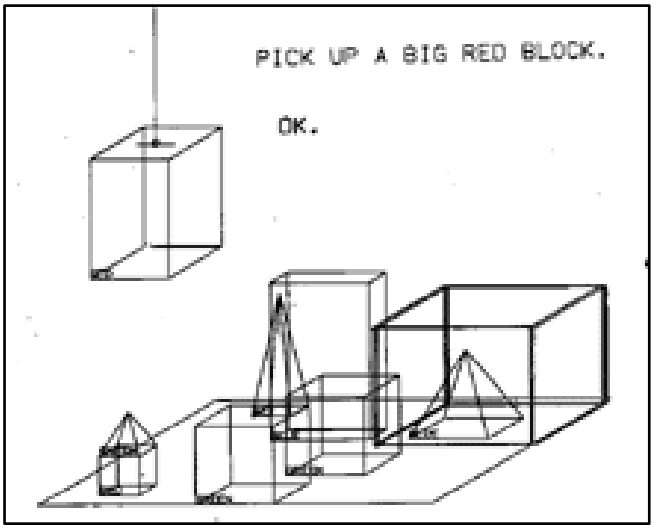
Slika 2.2. *Prikaz razgovora s Elizom*

Nekoliko godina nakon ELIZA-e razvijen je još jedan *chatbot* s fokusom na kliničku psihologiju, PARRY, koji se koristio za proučavanje šizofrenije. Osim ELIZA-inih regularnih izraza, PARRY sustav uključuje i model vlastitog mentalnog stanja s utjecajnim varijablama za razinu straha i ljutnje.

SHRDLU, program za razumijevanje prirodnog jezika kojeg je napisao Terry Winograd također na MIT-evom laboratoriju za umjetnu inteligenciju u razdoblju od 1968. do 1970. godine, bio je sofisticiraniji od prethodno opisanog programa ELIZA. SHRDLU vodi jednostavni dijalog s korisnikom o malom svijetu objekata (Svijet blokova) prikazanom na zaslonu. Tako korisnik šalje upite računalu o objektima, mijenjanju njihovih položaja te nazivanju pojedinih kolekcija, a računalo ih prevodi u fizičke akcije. Cijeli set blokova i lokacija je jednostavan i može se opisati s 50-tak riječi. Dio jednog takvog dijaloga prikazan je na Slici 2.3., a svijet blokova na Slici 2.4.

```
Person: PICK UP A BIG RED BLOCK.
Computer: OK. (does it)
Person: GRASP THE PYRAMID.
Computer: I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN.
Person: FIND A BLOCK WHICH IS TALLER THAN THE ONE YOU ARE HOLDING AND PUT IT INTO THE BOX.
Computer: BY "IT", I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING.
Computer: OK. (does it)
Person: WHAT DOES THE BOX CONTAIN?
Computer: THE BLUE PYRAMID AND THE BLUE BLOCK.
Person: WHAT IS THE PYRAMID SUPPORTED BY?
Computer: THE BOX.
Person: HOW MANY BLOCKS ARE NOT IN THE BOX?
Computer: FOUR OF THEM.
```

Slika 2.3. *Prikaz dijela dijaloga u SHRDLU-u*



Slika 2.4. *Originalni prikaz Svijeta blokova*

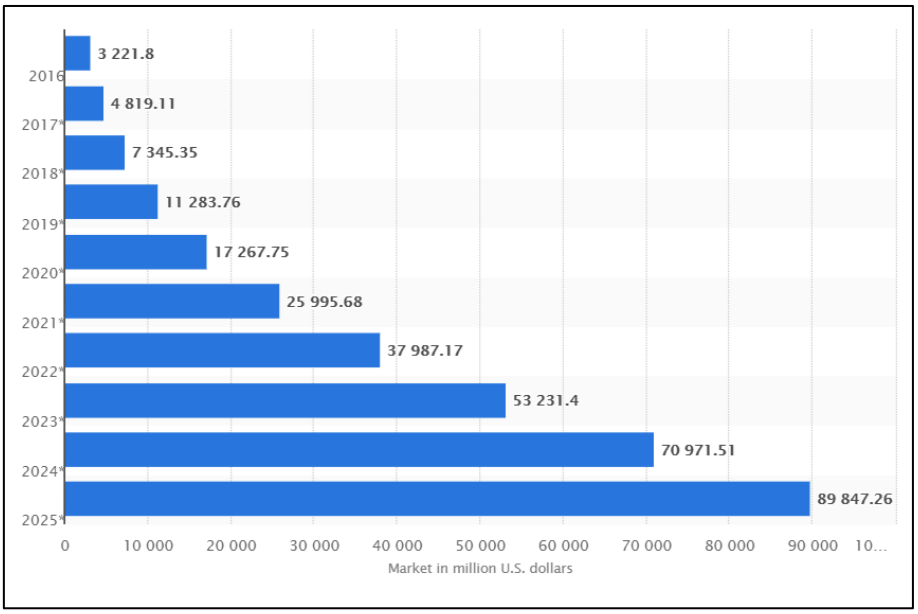
Opisano u [3] 70-ih i 80-ih godina umjetna inteligencija dobila je zamah. Zahvaljujući uspjesima u AI sustavima koji su tada razvijani na temama poput liječenja bolesti, uspostavljene su osnove koje se i dan danas koriste. AI se počinje koristiti u velikim projektima s praktičnim primjenama, a u svakom sljedećem projektu prilagođavana je rješavanju stvarnih problema. Najpoznatiji sustav tog vremena bio je MYCIN kojeg je izradio T. Shortliffe na Sveučilištu u Stanfordu. Sustav je korišten za dijagnosticiranje i terapijske odluke vezane za infektivne bolesti krvi i meningitis. Procjenom sustavnih odluka ocijenjeno je da je sustav dobar u donošenju odluka poput stručnjaka u tom području i puno bolji od ljudi koji nisu stručnjaci. Tako su u brojnim granama znanosti slijedili sljedeći sustavi: u medicini još PUFF (interpretacija podataka za testove pluća) i CADUCEUS (dijagnostika u internoj medicini), u kemiji DENDRAL (analiza molekularne strukture), u geologiji PROSPECTOR (analiza sastava kamenja), u računarstvu R1 (konfiguracija računala). Čak i u slučajevima kad su korisnički zahtjevi bili ostvareni tradicionalnim metodama, korištenje AI-a davalo bi širi pristup rješenjima zahvaljujući ekonomičnim programima i alatima.

Početkom 80-ih godina Japan je najavio ambiciozan Projekt pete generacije (engl. *Fifth Generation Project*) koji je dizajniran, između ostalog, da provodi praktički primjenjiva AI najsuvremenija istraživanja. Za razvoj AI-a Japanci su preferirali programski jezik PROLOG koji je 70-ih godina predstavljen kao europski duplikat američkog LISP-a. U PROLOG-u se određeni oblik predikatne logike mogao koristiti izravno kao programski jezik. Japan i Europa uglavnom su se koristili PROLOG-om, dok su se u Americi nastavili oslanjati na LISP. Sredinom 80-ih godina simbolički AI je dobio suparnika u neuronskim mrežama. Na temelju rezultata istraživanja mozga, McCulloch, Pitts i Hebb prvi su razvili matematičke modele za

umjetne neuronske mreže u 40-im godinama, ali tada nisu imali moćna računala. U 80-im godinama McCulloch-Pittsov neuron je doživio preporod u obliku tzv. konekcionizma. Konekcionizam je pristup kognitivnosti u kojem višestruke veze između čvorova (ekvivalentne moždanim stanicama) čine masivnu interaktivnu mrežu u kojoj se istodobno odvijaju mnogi procesi. Za razliku od simboličkog AI-a, konekcionizam je više usmjeren prema biološkom modelu mozga. Osnovna ideja je da se obrada podataka temelji na interakciji mnogih jednostavnih, jedinstvenih elemenata za obradu.

Neuronske mreže pružaju impresivnu izvedbu, osobito u području učenja. Netttalkov program bio je sposoban naučiti kako govoriti koristeći primjere rečenica: unosom ograničenog skupa pisanih riječi s izgovorom takva mreža mogla bi naučiti izgovoriti engleske riječi ispravno i ispravo primijeniti naučeno na nepoznate riječi. Ali, i opisani drugi pokušaj za neuronske mreže je došao prerano. Iako je bilo sredstava za financiranje, ograničenja su bila sveprisutna. Nije bilo dovoljno podataka za treniranje, nedostajala su rješenja za strukturiranje i modularizaciju mreža, a računala su i dalje bila jako spora. 1993. godine je na MIT-u razvijen Cog, humanoidni robot. 1997. godine IBM-ovo superračunalo Deep Blue pobijedilo je najpoznatijeg šahista, Garija Kasparova. Godinu poslije razvijen je Furby, prvi AI igrač. 2000. godine predstavljen je Kismet, robot koji koristi pokrete i mimiku u komunikaciji, a 5 godina poslije pokušava djelovati vlastitom moći uma.

2013. godine DeepMind prestavlja AI koji može igrati nekoliko Atari igara na razini ljudskih igrača, a dvije godine poslije uspješno igra 49 takvih igara. Pored klasičnih Atari igara, DeepMind je uspio 2015. godine poraziti najboljeg Go igrača na svijetu sa svojim AI AlphaGo. Što se tiče umjetne inteligencije danas, reklo bi se da je na vrhuncu i da doživljava svoje zlatno doba. Sve to je moguće zbog velikih količina podataka, brzih računala i značajnih napredaka u tehnikama strojnog učenja. Korištenjem neuronskih mreža nekoć komplicirani zadaci obrade videa i slika, analize teksta i prepoznavanja govora su sada rješivi, a postojeća rješenja s godinama će se još poboljšati. Do 2016. godine AI tržište doseglo je više od osam milijardi dolara, a pretpostavlja se da će do 2023. godine ta brojka iznositi oko pedeset. Na slici 2.5. je prikazana statistika AI tržišta prema [4] kroz nekoliko narednih godina.



Slika 2.5. *AI tržište*

U [5] se navodi da već postoje AI sustavi koji mogu nadmašiti ljude u određenim područjima, poput igranja GO-a ili analize podataka. No, još ne postoji opća umjetna inteligencija (engl. *Artificial General Intelligence*, skraćeno AGI) koja može raditi kao ljudsko biće, niti postoji superinteligencija koja je pametnija od ljudskog bića.

2.1.2. Strojno učenje

Strojno učenje (engl. *Machine learning*, skraćeno ML) programiranje je računala s ciljem optimiziranja kriterija uspješnosti pomoću poznatih primjera ili iskustva kao što je opisano u [2]. U strojnom učenju postoji model definiran na određene parametre. Model se uči ili trenira izvođenjem algoritma koji optimizira parametre. Modeli u strojnom učenju moraju moći predvidjeti svojstva novih podataka na temelju poznatih podataka, tj. moraju moći dobro generalizirati. Postoje tri bitna razloga za razvoj strojnog učenja:

1) postoje problemi presloženi za rješavanje pomoću algoritama, tj. ne postoji ideja kako ih riješiti (AI-potpuni problemi),
2) postoje sustavi koji se dinamički mijenjaju pa je potrebna prilagodba i
3) postoje ogromne količine podataka iz kojih se treba izvući znanje.

Podataka je puno i može ih se pronaći u raznim oblicima, a većina je javno dostupna i besplatna. S druge strane, znanje je ograničeno, tj. nema ga puno i skupo je. Cilj strojnog učenja je izgraditi modele koji interpretiraju podatke i iz njih izvlače zaključke. Za otkrivanje znanja u podacima koristi se dubinska analiza podataka (engl. *data mining*). To je primjena strojnog učenja na strukturirane i polustrukturirane baze podataka, a koristi se u brojim područjima: trgovini za

analizu potrošačke kartice, u financijama za određivanje kreditne sposobnosti i detekciju zlouporaba kartica, u proizvodnji za optimizaciju, u medicini za postavljanje dijagnoza, u telekomunikacijama za optimizaciju usluga i dr.

Strojno učenje dijeli se na nadzirano, nenadzirano i podržano/ojačano. U nadziranom učenju podaci su u obliku (ulaz, izlaz) = (x, y) i cilj je naći preslikavanje $y = f(x)$. Ako je y diskretna (nebrojčana) vrijednost radi se o klasifikaciji, a ako je y kontinuirana/brojčana vrijednost, radi se o regresiji [2, str. 19]. Regresija predstavlja pronalaženje najboljeg odgovarajućeg pravca, a udaljenost točaka od pravca označava količinu pogreške. Cilj u ovakvom modelu je da stroj nauči smanjiti grešku. Klasifikacija pak predstavlja svrstavanje koje daje odgovore "da" ili "ne" te "ovo" ili "ono". Npr.: „Je li ovo mačka?" Na što će odgovor biti "da" ili "ne" te "Je li ovo mačka ili pas?" Na što odgovor može biti "mačka" ili "pas". Kod nenadziranog učenja podaci nemaju ciljnu vrijednost, a pravilnost se nalazi grupiranjem, otkrivanjem novih vrijednosti ili smanjenjem dimenzionalnosti. Podržano/ojačano učenje bavi se radnjama koje se trebaju poduzeti u okruženju kako mi se maksimizirala ideja kumulativne nagrade.

Prema [6] nadzirani model prvo se mora trenirati na označenim primjerima nakon čega se može koristiti na neviđenim primjerima. Svaki neviđeni primjer predstavlja se vektorom značajki x. Za svaki ulazni primjer koji može biti visina, težina, prihod i dr., generira se vektor značajki x, tj. generira ga ekstraktor značajki. Za treniranje i za predikciju koristi se isti korak. U fazi treniranja model na ulazu dobiva primjere x i njihove oznake y, odnosno parove (x, y). Isti model se koristi i za predikciju gdje na ulazu dobiva samo primjere x, a na izlazu daje oznaku y.

Cilj je dobro generalizirati. Složeni model se može prilagoditi podacima i za njih davati jako dobre predikcije. No, ako je model previše složen, može se previše prilagoditi podacima na kojima je treniran i tako postaje prenaučen, tj. davat će loše predikcije na neviđenim podacima. Takav model je beskoristan jer ne generalizira dobro. Situacija se može izbjeći metodom unakrsne provjere koja ispituje koliko dobro će model raditi na neviđenim podacima.

Kako neviđeni primjeri nisu dostupni, dio postojećih primjera "glumi" neviđene primjere. Općenito, skup podataka dijeli se na skup za učenje i skup za testiranje. Omjer ta dva skupa je obično 70% / 30%. Model se uči, odnosno trenira, na skupu za učenje. Pomoću tog modela radi se predikcija na skupu za testiranje i na tom skupu se računa točnost. Točnost modela upućuje na sposobnost generalizacije modela jer model prethodno nije vidio podatke iz skupa za testiranje. Za određivanje optimalne složenosti modela potreban je treći skup podataka na kojem se računa točnost modela: skup za provjeru. Skup podataka se onda osim na skup za učenje i skup za testiranje dijeli još i na skup za provjeru (obično u omjerima 40%/30%/30%).

Modeli različite složenosti treniraju se na skupu za učenje te se svaki ispituje na skupu za provjeru. Nakon što se odabrao optimalni model na skupu za provjeru, točnost se izračunava na skupu za testiranje.

2.1.3. Duboko učenje

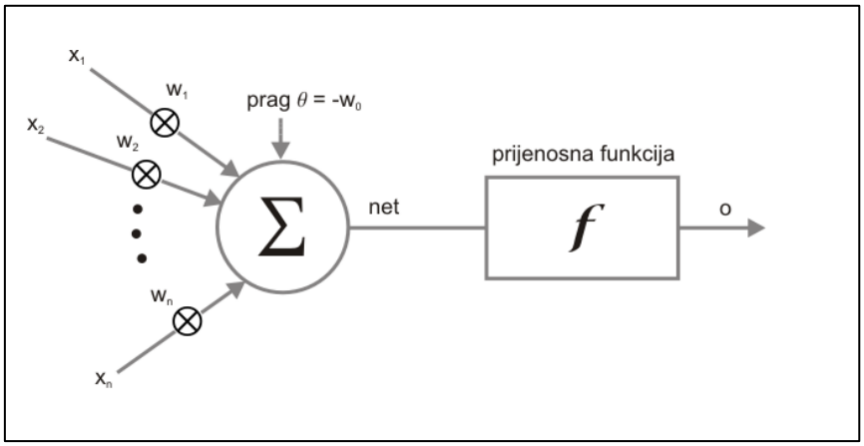
Prema [6] duboko učenje pristup je strojnog učenja koji se temelji na neuronskim mrežama i velikoj količini podataka. Najvažnija karakteristika neuronskih mreža je njihova sposobnost učenja od okoline. Proces učenja stalno se ponavlja i sastoji se od podešavanja sinaptičkih težina i pragova. Nakon svake iteracije mreža ima sve više znanja o okolini. U kontekstu neuronskih mreža, „učenje je proces kojim se slobodni parametri neuronske mreže prilagođavaju okolini u kojoj se mreža nalazi“ [7, str. 4]. Način na koji se parametri mijenjaju određuje vrstu učenja. Okolina dakle stimulira neuronsku mrežu, a kao rezultat te stimulacije neuronska mreža se mijenja i zbog promjena drukčije reagira na okolinu.

Neuronske mreže mogu se podijeliti na umjetne i duboke neuronske mreže kao što je navedeno u [8]. Umjetna neuronska mreža, u daljnjem tekstu ANN, (engl. *Artificial neural network*, skraćeno ANN) skup je međusobno povezanih jednostavnih procesnih elemenata. Funkcionalnost ANN temelji se na biološkom neuronu, a općenito služe za velike obrade podataka. Rad s ANN podrazumijeva fazu učenja i fazu obrade podataka. Učenje ili treniranje opetovani je postupak kojim se predočavaju ulazni primjeri, odnosno uzorci, i neki očekivani izlazi. Epoha predstavlja jedno predočavanje svih uzoraka. Postoje pojedinačno i grupo učenje. U pojedinačnom se za svaki primjer podešavaju faktori, a u grupnom učenju cijela epoha je u jednom opetovanju.

U težinama veza neurona pohranjeno je znanje o izlazu koje je funkcija ulaza. Postoje i dva načina učenja, a to su nadzirano, gdje postoje primjeri oblika (ulaz, izlaz), i nenadzirano učenje, gdje je izlaz nepoznat. Skup primjera za učenje dijeli se na tri skupa: skup za učenje, skup za testiranje i skup za provjeru. Skup za učenje opetovano podešava težine, skup za testiranje provjerava rad mreže, a skup za provjeru služi za konačnu provjeru.. Uvodi se mjera pogreške te se učenje provodi dok mreže ne daju odgovarajuću točnost obrade podataka.

Ljudski živčani sustav čini 10^{11} neurona. Ima ih 100 različitih vrsta, raspoređeni su po definiranom rasporedu, a svaki neuron povezan je s 10^4 drugih. Dijelovi neurona su soma, dendriti, akson i završni članci. Umjetni neuron analogan je biološkom neuronu. U modelu umjetnog neurona signali su numeričke vrijednosti, jakost sinapse opisuje se težinskim faktorom *w*, tijelo stanice predstavlja zbrajalo, a akson je prijenosna funkcija *f*. Signali koji putuju aksonom (npr. x_0) množe se sa sinaptičkim snagama dendrita (težinama) drugih neurona

(npr. w_0). Ideja je da se sinaptičke snage mogu mijenjati učenjem te određuju utjecaj jednog neurona na drugi. Svaki neuron ima aktivacijsku funkciju koja uzima sumu umnoška ulaza neurona s pripadnim težinama i praga (θ) te ih preslikava na izlaz neurona koji modelira signal. Umjetni neuron prikazan je Slikom 2.6.



Slika 2.6. *Umjetni neuron*

Ulazi su označeni sa x_1, x_2, ..., x_n a njihove pripadne težine sa w_1, w_2, ..., w_n, i prag sa θ za koji se zbog pojednostavljenja često uzima oznaka w_0. Izlaz neurona može se zapisati prema formuli (2-1).

$$y = f\left(\sum_{i=0}^{n} x_i\, w_i\right) \tag{2-1}$$

Postoje različite prijenosne funkcije (ANALINE, TLU, sigmoidalna jedinica) kao i moguće arhitekture (aciklička – mreža koja ide isključivo prema sljedećem sloju, mreža s povratnom vezom, lateralno povezana mreža).

Duboke neuronske mreže su umjetne neuronske mreže s više skrivenih slojeva između ulaznih i izlaznih slojeva. Duboke neuronske mreže pogodnije su za kompleksnije probleme klasifikacije. Svaki sloj mreže obrađuje podatke na drugoj razini apstrakcije i na temelju tih podataka daje neki izlaz. Kretanjem od ulaznog sloja prema izlaznom razina apstrakcije se povećava te se grade kompleksniji i apstraktniji koncepti odlučivanja. Ulazni sloj obrađuje podatke na razinama piksela dok izlazni sloj radi na najapstraktnijoj razini i daje rezultat klasifikacije. Većim brojem slojeva može se preciznije dekomponirati apstraktni problem klasifikacije na niz jednostavnih odluka koje se mogu donijeti na razinama piksela. S većim brojem slojeva ta je dekompozicija finija i preciznija.

2.2. Obrada prirodnog jezika

Prirodni jezik u svojim svim oblicima predstavlja najprirodniji oblik komunikacije među ljudima. Prema [9, 10] obrada prirodnog jezika, u daljnjem tekstu NLP, (engl. *Natural language processing*, skraćeno NLP) višedisciplinarno je područje koje se bavi interakcijom ljudskih i računalnih jezika, točnije kako programirati računala da obrađuju i analiziraju velike količine podataka u kojima je sadržan ljudski jezik. Glavna područja koja NLP pokriva su računarstvo i umjetna inteligencija, no također usko je povezano s lingvistikom, kognitivnim znanostima, psihologijom, filozofijom i matematikom, pogotovo logikom. Što se tiče računarstva, NLP se odnosi na teoriju formalnih jezika, tehnike prevođenja (engl. *compiler techniques*), dokazivanje teorema, strojno učenje i ljudsko-računalnu interakciju (engl. *Human–computer interaction*, skraćeno HCI). Općenito, problemi kojima se NLP bavi su prepoznavanje jezika, razumijevanje prirodnog jezika (engl. *Natural language understanding*, skraćeno NLU) te generiranje prirodnog jezika (engl. *Natual language generation*, skraćeno NLG).

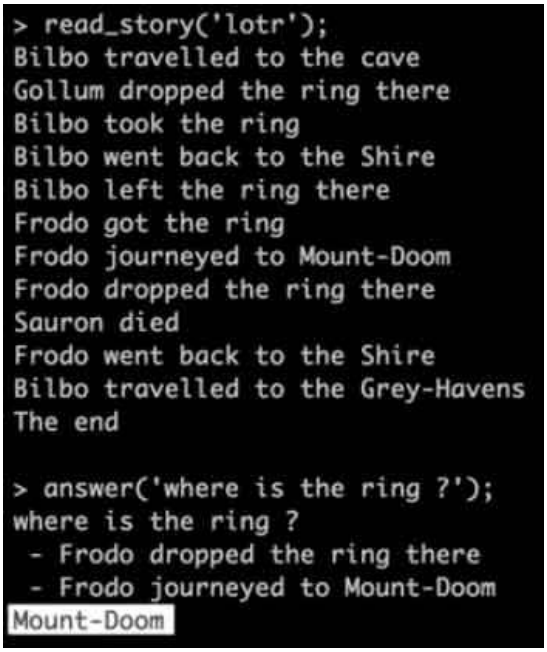
Razvoj NLP-a počinje razvojem umjetne inteligencije sredinom prošlog stoljeća. Ranije navedeni Turingov test, SHRDLU i ELIZA sustavi su s integriranim NLP-em. 70-ih godina programeri su počeli pisati tzv. „konceptualne ontologije“ koje su strukturirale informacije iz stvarnog svijeta u podatke razumljive računalu. 80-ih godina sustavi za obradu prirodnog jezika bazirali su se na kompleksnim rukom pisanim pravilima, a u kasnijim 80-im godinama dogodila se revolucija NLP-a predstavljanjem strojnog učenja i algoritama za obradu jezika. 1982. godine predstavljen je koncept *chatbota* i počeo je Jabberwacky program. Ideja projekta bila je kreirati AI program koji bi mogao simulirati prirodni jezik u zanimljivom, zabavnom i humorističnom smislu, a sve kako bi mogao proći Turingov Test. Ovo je bila još jedna primjena NLP-a i dodatno je proširila zajednicu istraživača i prilika za ulaganje novčanih sredstava u to područje. 90-ih godina došlo je do velikog napretka NLP-a i razvoja statističkog NLP-a. Razvoj i javno korištenje Interneta omogućili su začetak strojnog učenja i samim time strojnog prevođenja. Značajan napredak očitovao se u korištenju jezika i prepoznavanja jezika zbog novih informacija i tekstova koje su računala mogla čitati. U 21. stoljeću istraživanja i razvoj NLP-a doživljavaju eksploziju. Računalna snaga i memorija su jeftini i svima dostupni. NLP doživljava transformaciju zahvaljujući ogromnim informacijama koje se mogu lako pohraniti na modernim računalima te su prethodnoj statističkoj obradi dodana gramatička pravila. Samim time, otvaraju se nove teme: rangiranje strojnog čitanja teksta, projekt u kojem računala mogu čitati, sažeti i razumjeti tekst do prepoznavanja jezika Appleove Siri, Amazonove Alexe ili Microsoftove Cortane. Moderni NLP sastoji se od prepoznavanja jezika, strojnog učenja,

strojnog čitanja teksta i strojnog prevođenja. Kada bi se to ujedinilo, umjetna inteligencija bi dobila stvarno znanje o svijetu, a ne bi morala samo igrati šah ili prelaziti preko nekih prepreka. U skorijoj budućnosti računala će moći čitati informacije s Interneta i učiti iz njih te rješavati probleme i možda liječiti bolesti. Krajnji cilj za NLP i AI je dosegnuti razinu svijesti i razumijevanja ljudskog bića.

2.2.1. Pristupi

Prema [11] Razine lingvističke analize sastoje se od sintakse, semantike i pragmatike. Sintaksa podrazumijeva gramatiku, semantika značenje, a pragmatika smisao. Analogno za programske jezike sintaksa bi značila da nema greške pri prevođenju, semantika da nema implementacijskih *bugova*, a pragmatika da je implementiran ispravni algoritam. Nadalje će se objasniti četiri pristupa prilikom obrade prirodnog jezika, a to su distribucijski pristupi, pristupi bazirani na okvirima, model-teorijski pristupi te interaktivno učenje.

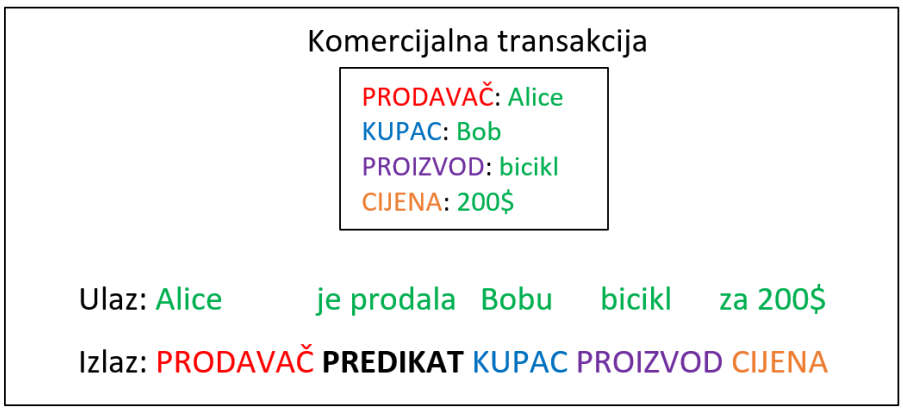
Distribucijski pristupi uključuju velike statističke metode strojnog i dubokog učenja. Te metode obično pretvaraju sadržaj u vektore riječi za matematičku analizu i prilično dobro se izvode u zadacima kao što su označavanje dijela jezika (je li ovo imenica ili glagol), analiza ovisnosti riječi u rečenici (mijenja li ovaj dio rečenice drugi) i semantička povezanost (koriste li se ove različite riječi na sličan način). Opisani NLP zadaci ne oslanjaju se na razumijevanje značenja riječi, nego na odnos između samih riječi. Takvi sustavi su široki, fleksibilni i skalabilni te se mogu primijeniti na različite tipove teksta bez potrebe za ručnim projektiranjem značajki ili stručnim znanjem u području kodiranja. Nedostatak je što im nedostaje razumijevanje stvarne semantike i pragmatike. Usporedba riječi s drugim riječima, ili riječi s rečenicama, ili rečenica s rečenicama može dati različite ishode. Semantička sličnost, npr., ne znači sinonimiju, a proračunom najbližeg susjeda sustav može smatrati da su antonimi povezane riječi. Napredni moderni modeli neuronskih mreža, kao što su memorijske mreže s kraja na kraj koje je pokrenuo Facebook ili združeni model više zadataka kojeg je izumio Salesforce, mogu se nositi s jednostavnim zadacima tipa pitanje-odgovor, ali su još u ranim fazama razvoja za potrošačku i poduzetničku primjenu. Dosad je Facebook javno pokazao da neuronska mreža trenirana na apsurdno pojednostavljenu verziju Gospodara prstenova može zaključiti gdje se nalazi nedostižni Prsten. Treniranje i testiranje neuronske mreže na opisanom primjeru vidljivo je na Slici 2.7.



Slika 2.7. *Treniranje i testiranje neuronske mreže*

Distribucijske metode postižu širinu, ali ne i dubinu. Potrebno je još pronaći zadovoljavajuće odgovore na složena i nijansirana pitanja koja se oslanjaju na lingvističku sofisticiranost i kontekstualno znanje o svijetu.

Sljedeći pristup je pristup baziran na okvirima. „*Okvir je podatkovna struktura za predstavljanje stereotipne situacije*“, objašnjava Marvin Minsky u svom radu 1974. godine pod nazivom „Okvir za predstavljanje znanja“ (engl. *A Framework For Representing Knowledge*). Okviri se mogu zamisliti kao kanonsko predstavljanje za koje se pojedinosti mogu izmjenjivati. Kao primjer okvira može poslužiti komercijalna transakcija. U takvim situacijama obično postoje prodavači, kupci, roba koja se razmjenjuje i cijena razmjene. Na Slici 2.8. grafički je prikazan primjer.



Slika 2.8. *Primjer okvira za komercijalnu transakciju*

Rečenice koje su sintaktički različite, ali semantički iste, kao što su "Alice je prodala Bobu bicikl za 200 dolara" i "Bob je od Alice kupio bicikl za 200 dolara", mogu se uklopiti u isti okvir. Raščlanjivanje tada uključuje prepoznavanje okvira koji se koristi, a zatim popunjavanje specifičnih parametra okvira, npr. Alice, 200 dolara. Očiti nedostaci opisanih okvira su zahtijevanje nadzora. U nekim područjima stručnjak ih mora kreirati što ograničava područje pristupa baziranih na okvirima. Okviri su također nužno nepotpuni. Rečenice poput "Alice je posjetila trgovinu s biciklima jučer" i "Alice je kupila najjeftiniji bicikl" ne mogu se adekvatno analizirati prije navedenim okvirom.

Treća kategorija semantičke analize je model-teorijski pristup. Za razumijevanje ovog pristupa predstavit će se dva važna lingvistička koncepta: teorija modela i kompozicionalnost. Teorija modela odnosi se na ideju da se rečenice odnose na svijet, kao u slučaju s utemeljenim jezikom (npr. blok je plavi). U kompoziciji, značenja dijelova rečenice mogu se kombinirati kako bi zaokružili cijelo značenje. Ovaj pristup se može usporediti s uključivanjem jezika u računalne programe. Da bi se otkrio odgovor na pitanje „Koji je najveći grad u Europi po broju stanovnika" prvo se moraju identificirati pojmovi „grad" i „Europa", pretražiti gradovi u Europi, poredati brojevi stanovnika za svaki grad koji je ušao u uži izbor i vratiti najveću vrijednost. Da bi se izvršila rečenica „Podsjeti me kupiti mlijeko nakon mog posljednjeg sastanka u ponedjeljak" potrebno je slična dekompozicija i rekombinacija.

Modeli se dijele na one kojima je potreban stručni nadzor i na one koje mogu nadzirati prosječni ljudi. Prednosti pristupa temeljenim na modelima uključuju predstavljanje cijelog svijeta, bogatu semantiku i obradu s kraja na kraja koja omogućuje davanje odgovara na teška pitanja i nijansirane upite. Glavni nedostatak je da su aplikacije ograničene opsegom zbog potrebe za ručnim projektiranjem značajki. Primjena pristupa temeljenih na teorijskima modelima u razumijevanju prirodnog jezika obično počinje od najlakših i najčešćih slučajeva pa sve do težih. Idealni pristup za razumijevanja prirodnog jezika bio bi i širina i dubina, ali u praksi je ili jedno ili drugo. Distribucijske metode imaju razmjere i širinu, ali plitko razumijevanje. Pristupi bazirani na teorijskim modelima traže intenzivan rad te su uske u opsegu, dok metode bazirane na okvirima se nalaze negdje između.

Posljednja kategorija semantičke analize je interaktivno učenje. Paul Grice, britanski filozof jezika opisao je jezik kao suradničku igru između govornika i slušatelja. Kako bi se održali pristupi dubine i širine u učenju jezika, potrebno je upotrijebiti dinamična i interaktivna okruženja u kojima ljudi postupno uče računala. Takvi pristupi bi mogli omogućiti računalima rješavanje NLP i NLU problema s kraja na kraj bez eksplicitnih modela.

2.2.2. Razumijevanje prirodnog jezika

Razumijevanje prirodnog jezika, u daljnjem tekstu NLU, (engl. *Natural language understanding*) grana je umjetne inteligencije koja koristi računalni softver za razumijevanje ulaznih varijabli u obliku tekstualnih rečenica ili govornom formatu kao što je u [11]. Uključuje modeliranje ljudske spoznaje čitanja ili, drugim riječima, NLU analizira i prevodi ulazne varijable prema načelima prirodnog jezika. Jednostavan način za razumijevanje NLU-a je gledati na dostupne usluge potrošača i poslovnih proizvoda koji modeliraju NLU. Npr., Appleova Siri ili Amazonova Alexa izvode NLU procese u kontaktu slušanja i dešifriranja korisničkih ulaznih zahtjeva. Sličan NLU ugrađen je u Amazonov "Lex", poduzetničku uslugu za izradu platformi za strojno učenje. Razumijevanjem kako se NLU primjenjuje u tim aplikacijama, lako je vidjeti da NLU uključuje razumijevanje jezičnih inputa.

Razumijevanje prirodnog jezika široko je rasprostranjeno u akademskoj zajednici i industriji. Primjenjuju se u glasom kontroliranim asistentima (Siri, Google Now, Microsoft Cortana), pretraživanju prirodnog jezika (Google, Facebook Graph Search), odgovaranju na pitanja (Google, IBM's Watson, Wolfram Alpha), analizi za automatsko trgovanje, poslovnoj inteligenciji (Palantir, Quid), analitici društvenih mreža, sažimanju sadržaja (Summly) i drugdje.

NLU dakle pokušava razumjeti značenje pisanog teksta. Npr., nakon što softver za prepoznavanje govora pretvori govor u tekst, NLU je na redu za dešifriranje značenja toga govora. Moguće je da isti tekst ima različita značenja, da različite riječi imaju isto značenje ili da se značenje mijenja s kontekstom. Neki od izazova s kojima se suočavaju NLU sustavi su poznavanje pravila i strukture jezika te razumijevanje teksta bez dvosmislenosti. Google-ov API.ai pruža alate za prepoznavanje govora i NLU.

Nevezano za pristup koji se koristi, većina sustava za razumijevanje prirodnog jezika dijeli neke zajedničke komponente. Sustav treba leksikon jezika te raščlanjivač sintakse i gramatičkih pravila za rastav rečenice na sastavne dijelove. Sustav također treba i semantičku teoriju koja će provoditi razumijevanje. Interpretacijske sposobnosti NLU-a ovise upravo o semantičkoj teoriji koju koristi. Općenito, semantičke teorije kreću se od onih s naivnom semantikom ili stohastičkom semantičkom analizom do onih koje koriste pragmatičnost za dobivanje značenja iz konteksta.

2.2.3. Generiranje prirodnog jezika

Generiranje prirodnog jezika, u daljnjem tekstu NLG, (engl. *Natural language generation*) proces je kojim se misao prenosi na jezik [13]. Proučavali su ga filozofi, neurolozi, psiholozi, lingvisti i dr. Ovdje se radi o generiranju sa stajališta računalne perspektive, tj. umjetne inteligencije i računalne lingvistike. NLG alati analiziraju podatke, interpretiraju ih, pronalaze najznačajnije dijelove i generiraju izvješća. U osnovi, NLG unosi umjetnu inteligenciju u poslovnu inteligenciju (engl. *Business intelligence*, skraćeno BI), automatizira rutinsku analizu te štedi vrijeme i novac poslovnih korisnicima. Iako BI proizvodi generiraju vizualizacije, izvješća i kontrolne ploče, poslovni korisnici i dalje moraju analizirati i tumačiti podatke. Tu na snagu dolazi NLG. Automatski provodi analizu i generira prijevode za ono što je značajno i smisleno u podacima. Poslovni korisnici više ne moraju proučavati podatke kako bi interpretirali njihovo značenje jer NLG alati to čine umjesto njih. Štoviše, NLG alati zaobilaze potrebu za stvaranjem vizualizacija, grafikona i izvješća. Alati prolaze kroz velike količine podataka i generiraju izvješća automatski.

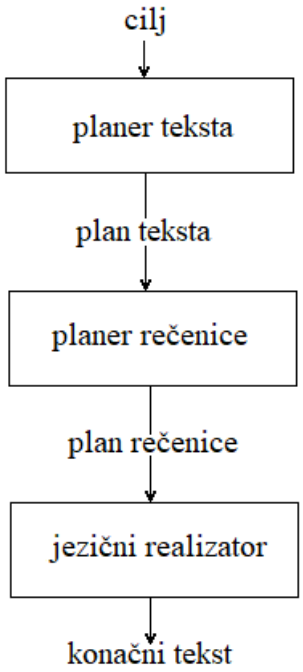
Prema s gledišta NLG sustava, „generator“ (ekvivalent osobe koja ima nešto za reći) je računalni program. Proces generiranja započinje početnom namjerom za komuniciranjem, a potom se vrši određivanje sadržaja onoga što će se reći, odabir riječi i retoričke organizacije te prilagodba gramatici oblikovanjem riječi pisanog teksta ili uspostavljanje prozodije[2] govora. Danas, ono što generator generira može biti u rasponu od jedne riječi ili fraze dane u odgovoru na pitanje do više rečenica u dijalogu s primjedbama i pitanjima, s objašnjenjima na više stranica ovisno o kapacitetu i ciljevima programa za koji se generira i čiji zahtjevi i pojedinosti se uzimaju u obzir.

U [15] proces generiranja dijeli se na tri faze:

1) identificiranje ciljeva izjave,
2) planiranje kako se ciljevi mogu postići procjenom stanja dostupnih komunikacijskih resursa i
3) realiziranje planova u tekst.

Opisane tri faze prikazane su na Slici 2.9., a uključuju sintaktičku, morfološku i pravopisnu obradu.

[2] prozodija (grč. *prosodia*) dio metrike: izgovaranje riječi po naglasku (akcentu) u dužini slogova; pravila o dužini i kratkoći slogova prema [14]



Slika 2.9. *Arhitektura NLG sustava*

Problemi s kojima se NLG najčešće susreće su određivanje sadržaja, tj. odlučivanje koje informacije se trebaju uključiti u tekst, a koje izostaviti, struktura dokumenta, tj. odlučivanje kako će se tekst organizirati i strukturirati, leksikalizacija, tj. odabiranje pojedinih riječi ili fraza koje su potrebne za interpretiranje navedenih podataka, agregacija, tj. odlučivanje kako se informacije trebaju sastaviti u rečenične dijelove, referentno generiranje izraza, tj. određivanje svojstava entiteta koji bi se trebali koristiti kada se govori o tom entitetu te obrada površine, tj. utvrđivanje kako se temeljni sadržaj teksta treba pretvoriti u niz gramatički ispravnih rečenica koje izražavaju željeno značenje.

2.3. NLP servisi

Kombinacija NLP i NLU tehnologija postaje sve bitnija u raznim softverskim područjima uključujući i *chatbot* tehnologije. Mnoga poduzeća su fokusirana na opisane tehnologije, a ovdje će se predstaviti tri najpopularnija NLP servisa prema istraživanju [16].

2.3.1. Dialogflow (Api.ai)

Api.ai je danas poznat kao Dialogflow. Razvio ga je Google 2016. godine. Ova platforma se povezuje s korisnicima na Google asistentu ili drugim aplikacijama i uređajima za izmjenjivanje poruka. Glavne karakteristike NLP-a i ML-a ove platforme uključuju milijune dijaloških procesa koji se obrađuju svaki dan, tehnike za treniranje na postojećim podacima te cijeli tijek stvaranja dijaloga. DialogFlow koristi intente, entitete, akcije s parametrima, kontekste te govor u tekst i tekst u govor sposobnosti, kao i strojno učenje koje tiho radi i trenira model. Ima ugrađeno znanje o temama kao što su neformalni razgovori, vrijeme i mudrost što znači da se agent ne mora trenirati za te svrhe. DialogFlow vraća izlaz kao JSON podatak. Programski jezici, aplikacije i integracije koji se mogu koristiti su SDK za Android, iOS, Cordova, JavaScript, HTML, Node.js, .NET, Unity, Xamarin, C++, Python, Ruby, PHP (Community Supported), Epson Moverio, Botkit, JAVA. Postoji ugrađena integracija za Google Assistant, Facebook Messenger i Slack. Najpoznatija platforma je Google Assistant, a moguće je koristiti i na drugom platformama za izmjenjivanje poruka. Podržava brojne svjetske jezike: brazilski portugalski, kineski, engleski, nizozemski, francuski, njemački, talijanski, japanski, korejski, portugalski, ruski, španjolski i ukrajinski. Besplatan je te nema ograničen broj API poziva. Ima omogućenu integraciju treće strane što znači da je dopušten u servisima za izmjenjivanje poruka i dr. Prigodan je za srednje razine B2C *chatbotova*, virtualne asistente i MVP.

2.3.2. Wit.ai

Wit.ai je sada Facebookova SaaS platforma koja programerima olakšava izgradnju *chatbota* za aplikacije ili uređaje. Može se koristiti na platformama koje uče nove naredbe semantički, ali i na onim platformama gdje programer unosi naredbe. Programski jezici, aplikacije i integracije koje se mogu koristiti su Node.js klijent, Python klijent, Ruby klijent, na drugim platformama HTTP API. Općenito, platforme na kojima se može koristiti su aplikacije i uređaji na kojima se može razgovarati ili razmjenjivati poruke. Podržava oko 50 jezika, a među njima i hrvatski. Besplatan je te također ima neograničen broj API poziva. No, ne podržava integraciju

trećih strana. Prikladan je za jednostavne B2C *chatbotove* i MVP. Wit.ai ima ugrađen NLP i detektira 15-tak entiteta.

2.3.3. IBM Watson

Watson je AI platforma koju je izradio IBM. Može razumjeti sve oblike podataka, biti u interakciji s ljudima i učiti iz tih interakcija. IBM je ovu tehnologiju premjestio na oblak i objavio API omogućujući korisnicima izrađivanje *chatbotova*. Izrađen je na neuronskim mrežama, a glavne komponente od kojih se sačinjen su intenti, entiteti i dijalozi. Programski jezici, aplikacije i integracije koje podržava su Node SDK, Java SDK, Python SDK, iOS SDK, Unity SDK. Može se koristiti na bilo kojoj platformi za razmjenjivanje poruka koje podržavaju govor, slike i tekst. Podržava dva jezika, engleski i japanski. Nije besplatan te se plaćanje određuje prema broju upita, API poziva ili zahtjeva. Ovisno o planu plaćanja, ima ograničen broj API poziva.. Od integracija treće strane podržava IBM servise. Prikladan je za virtualne asistente i *chatbotove* koji su integrirani u IBM servise.

3. CHATBOT SUSTAVI

Prema [17] *chatbot* sustavi ili *chatbotovi* su sustavi koji mogu vršiti razgovore s ciljem oponašanja nestrukturiranih razgovora ili razgovora karakterističnih za međuljudsku interakciju. Često imaju zabavnu vrijednost, kao Microsoftov „XioaIce“ sustav koji razgovara s ljudima na platformama za izmjenjivanje poruka. Počevši od prvog sustava, ELIZA, *chatbotovi* su korišteni za praktične svrhe kao što su testiranje teorija psihološkog savjetovanja. Kao i sve ostalo u jezičnoj obradi, *chatbot* arhitekture mogu se svrstati u dvije klase: sustavi temeljeni na pravilima i sustavi temeljeni na korpusu. Sustavi koji se temelje na pravilima uključuju rane utjecaje ELIZA i PARRY sustava. Sustavi temeljeni na korpusu uče iz velikih skupova podataka u kojima su sadržani ljudski razgovori tako da na korisnikov upit mogu dati odgovor. Sustavi uče koristeći pronalaženje informacija, u daljnjem tekstu IR sustavi (engl. *Information Retrieval*, pronalaženje informacija, skraćeno IR), gdje jednostavno kopiraju ljudski odgovor iz prethodne komunikacije ili koriste paradigmu strojnog prevođenja kao što su neuronske mreže slijeda sekvenci.

3.1. Sustavi temeljeni na pravilima

U pristupu koji se temelji na pravilima, *chatbot* odgovara na pitanja koja se temelje na nekim pravilima na kojima je treniran. Definirana pravila mogu biti vrlo jednostavna, ali i vrlo složena. Stvaranje tih *chatbota* relativno je jednostavno pomoću nekog pristupa koji se temelji na pravilima, ali *chatbot* nije učinkovit u odgovaranju na pitanja čiji se uzorak ne podudara s pravilima na kojima je treniran. Jedan od takvih jezika je AIML (engl. *Artificial Intelligence Markup Language*, jezik za označavanje umjetne inteligencije). Temelji se na XML-u koji omogućuje razvojnim programerima pisanje pravila za *chatbotove*. Također, pisanje pravila za različite scenarije oduzima puno vremena i nemoguće je napisati pravila za svaki mogući scenarij. Takvi *chatbotovi* mogu se nositi s jednostavnim upitima, ali se ne snalaze sa složenima pa tako nikad neće moći proći Turingov test.

Najbolji primjeri sustava temeljenim na pravilima su ranije spomenuti ELIZA i PARRY *chatbotovi*. ELIZA je najbitniji *chatbot* sustav u povijesti ovog područja. Dizajniran je da simulira Rogerovog psihologa koji inače provodi „terapiju usmjerenu na klijenta“, tj. takve metode ocrtavaju pacijenta odražavajući pacijentove izjave natrag na njih. Ako pacijent kaže „Otišao sam na dugu vožnju brodom.“, a psihijatar kaže „Reci mi o brodovima.“, ne pretpostavlja se da pacijent ne zna što je brod, već psihijatar s ovim pitanjem ima neki cilj komunikacije.

Nekoliko godina nakon ELIZA-e razvijen je još jedan *chatbot* s fokusom na kliničku psihologiju, PARRY, koji se koristio se za proučavanje shizofrenije. Osim ELIZA-inih regularnih izraza, PARRY sustav uključuje i model vlastitog mentalnog stanja, s utjecajnim varijablama za razinu straha i ljutnje. Neke teme razgovora mogu dovesti PARRY-ja da postane ljući ili nepovjerljiviji. Ako je PARRY-ova varijabla ljutnje visoka, odabire odgovore iz skupine "neprijateljskih" odgovora. Ako mu korisnik da do znanja da je neprijateljski nastrojen, PARRY povećava vrijednost varijable straha i postaje paranoidan. U siječnju 1973. godine kao demonstracija na međunarodnoj računalnoj konferenciji koristeći ARPANET ostvarena je komunikacija između *chatbota* ELIZA i PARRY. Na Slici 3.1. je prikazan dio njihovog razgovora.

ELIZA: Good Evening. Tell me your problems. Please terminate input with a period or a question mark.

PARRY: People get on my nerves sometimes.

ELIZA: I am not sure I understand you fully.

PARRY: You should pay more attention.

ELIZA: Suppose you should pay more attention.

PARRY: You're entitled to your own opinion.

ELIZA: What makes you think I am entitled to my own opinion?

PARRY: It bothers me just to be around people in general.

ELIZA: Please go on.

PARRY: I went to the track at Bay Meadows a while back.

ELIZA: What does that suggest to you?

PARRY: I went to the races.

Slika 3.1. *Razgovor između ELIZA i PARRY chatbotova*

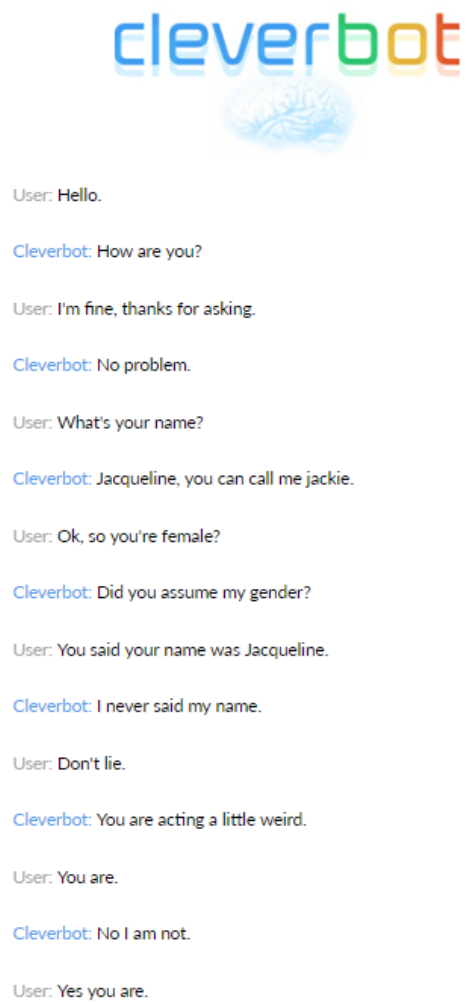
3.2. Sustavi temeljeni na korpusu

Sustavi temeljeni na korpusu[3], za razliku od sustava temeljenim na pravilima, stvaraju komunikaciju na temelju ljudskih razgovora. Takvi korpusi su u velikim količinama dostupni na *chat* platformama, na Twitteru, u filmskim dijalozima i pokazuju sličnost s prirodnim razgovorima. *Chatbot* odgovori se mogu izdvojiti i iz rečenica u korpusu ne-dijaloškog teksta. Postoje dva tipa sustava temeljena na korpusu i to su sustavi bazirani na pronalaženju informacija i sustavi temeljeni na nadziranom strojnom učenju koje uključuje slijed sekvenci (engl. *sequence-to-sequence*). Većina sustava bazirana na korpusu usredotočuje se na stvaranje odgovora koji su primjereni s obzirom na neposredno postavljen upit korisnika i zbog toga se često nazivaju sustavi generiranja odgovora. *Chatbotovi* bazirani na opisanom sustavu slični su sustavima za odgovaranje koji se fokusiraju na jedinstvene odgovore, a ignoriraju kontekst ili dublji cilj razgovora.

3.2.1. IR sustavi

Načelo koje stoji iza *chatbota* temeljenih na pronalaženju informacija je odgovoriti na korisnikovo pitanje X ponavljanjem nekih prikladnih pitanja i odgovora Y iz korpusa C prirodnog (ljudskog) teksta. Razlike u takvim sustavima leže u tome kako odabiru korpus i kako odlučuju što se računa kao ljudski prikladni odgovor za kopirati. Pristupi kod korpusa su prikupljati baze ljudskih razgovora koje mogu doći s raznih društvenih platformi ili koristiti korpus filmskih dijaloga. Nakon što se *chatbot* pusti u praksu, odgovori koje ljudi daju u komunikaciji s *chatbotom* se mogu uzeti kao dodatni podaci za treniranje. Uzimajući u obzir korpus i korisnikovu rečenicu, IR sustavi mogu koristiti jedan do dva algoritma za pronalaženje odgovora: vratiti odgovor na najsličnije pitanje ili vratiti najsličniji odgovor. Ideja kod prvog je tražiti pitanje u korpusu C koje je najsličnije korisnikovom pitanju i vratiti ljudski odgovor na to pitanje. Ideja kod drugog slučaja je direktno spojiti korisnikovo pitanje s odgovorom iz korpusa C, budući da dobar odgovor često dijeli riječi ili semantiku s prethodnim pitanjem. U oba slučaja mogu se koristiti bilo koje funkcije sličnosti. Iako se vraćanje odgovora na najsličnije pitanje čini kao intuitivni algoritam, vraćanje najsličnijeg odgovora u praksi prolazi bolje budući da ne dodaje još jedan sloj indirektnosti. Komercijalne implementacije IR pristupa uključuju Cleverbot i Microsoftov 'XioaIce'. Na Slici 3.2. prikazan je dio razgovora s Cleverbotom, a *chatbot* je dostupan na web stranici http://www.cleverbot.com/.

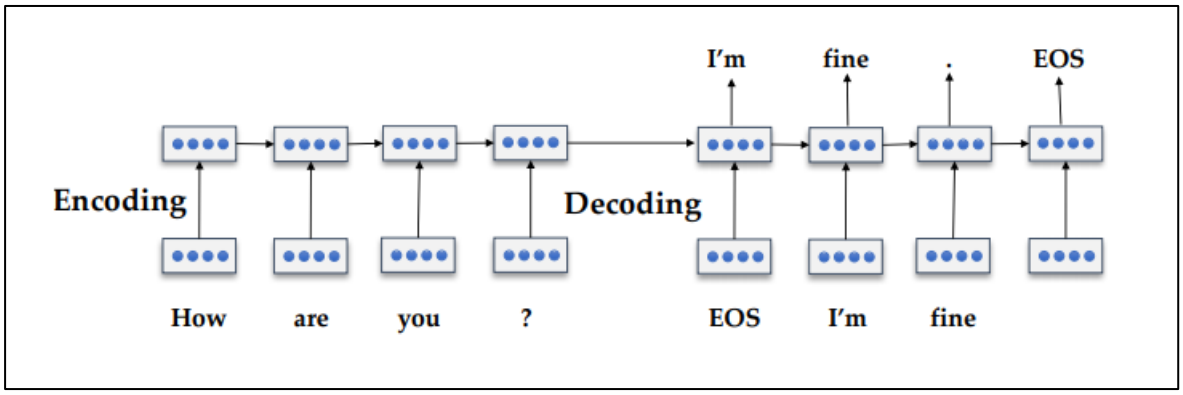
[3] Korpus (lat. corpus) tijelo, cjelina, ukupnost, skup, u ovom slučaju skup koji se sastoji od riječi, rečenica, fraza, izvučenih iz ljudskih razgovora, filmskih dijaloga ili drugih izvora prema [14]



Slika 3.2.: *Razgovor s Cleverbotom*

3.2.2. Sustavi slijeda sekvenci

Alternativni način korištenja korpusa za generiranje dijaloga je smatrati generiranje odgovora kao zadatak pretvaranja korisnikovog prethodnog upita u odgovor sustava. Ovo je u osnovi ELIZA sustav u verziji strojnog učenja: stroj uči iz korpusa kako pretvoriti pitanje u odgovor. Ova ideja prvo je razvijena koristeći strojno prevođenje bazirano na prijevodu fraza za prijevod korisničkih upita u odgovore sustava. Ubrzo je postalo jasno da se generiranje odgovora previše razlikuje od strojnog prevođenja. U strojnom prevođenju riječi ili fraze u izvornoj i ciljnoj rečenici se međusobno poklapaju, ali u razgovoru, korisnička izjava (odgovor) ne dijeli uvijek riječi ili izraze sadržane u prethodno postavljenom pitanju. Opisani modeli za generiranje odgovora modelirani su koristeći model slijeda sekvenci prikazan na Slici 3.3. Ukratko, postoji ulazna sekvenca u kojoj je postavljeno pitanje „How are you?“ i koja se kodira. Dekodiranjem se dobiva odgovor „I'm fine.“, a odgovor je dobiven pretvaranjem ulazne sekvence, tj. neuronskim strojnim prevođenjem.



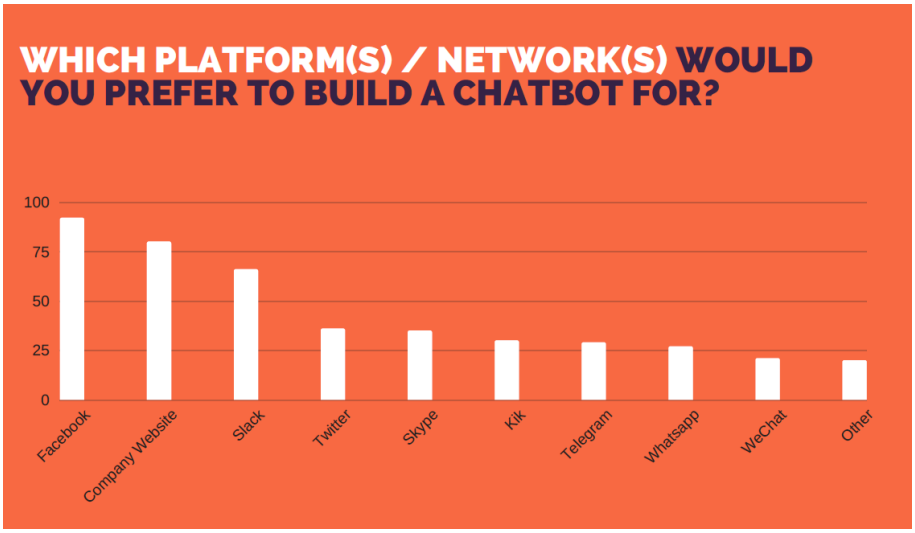
Slika 3.3. *Sustav slijeda sekvenci*

Potreban je određen broj modifikacija ovog modela da bi se prilagodio zadatku generiranja odgovora. Npr., osnovni modeli slijeda sekvenci imaju tendenciju proizvesti predvidljive, ali ponavljajuće i stoga suhoparne odgovore kao „Dobro sam“ i „Ne znam“ koji ograničavaju razgovor. To se može riješiti zamjenom objektivne funkcije za trening modela sa zajedničkim informacijskim ciljem ili modificiranjem slijeda dekodera da održava različite odgovore u slijedu. Drugi problem ovog modela je nemogućnost modeliranja duljeg prethodnog konteksta razgovora. To se može riješiti dopuštajući modelu da vidi prethodne odgovore, kao što je korištenje hijerarhijskih modela koji sažimaju informacije prethodnih nekoliko odgovora. Modeli za generiranje odgovora korištenjem slijeda sekvenci usredotočuju se na pojedinačne odgovore i zato ne rade dobar posao kontinuiranog generiranja odgovora koji se nadovezuju jedan na drugi. To se pak može riješiti korištenjem tzv. „pojačanog učenja“ kao i tehnika kao što su „suparničke mreže“ kako bi model naučio odabrati odgovore koji bi cijeli razgovor učinili što prirodnijim.

3.3. Trenutno stanje chatbot industrije

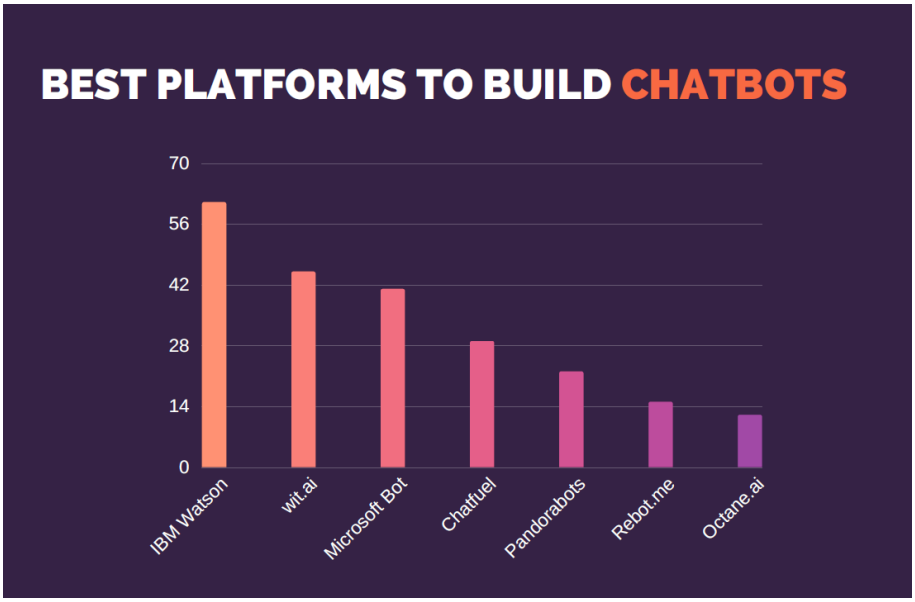
U razdoblju od studenog 2016. do siječnja 2017. provedeno je istraživanje koje se fokusiralo na trenutno stanje *chatbota* i njihove izglede u budućnosti. U istraživanju su sudjelovali pojedinci iz širokog spektra industrije: tehnologije, edukacije, marketinga, oglašavanja, logistike, online trgovine, avijacije, proizvodnje, ugostiteljstva i dr.. Opisano istraživanje proveo je Mindbowser u suradnji s ChatbotsJournal.com. Prema [18] rezultati su sljedeći: Mindbowser je glavni davatelj *chatbot* usluga u više industrijskih grana. ChatbotsJournal.com je najaktivnija mrežna zajednica *chatbot* stručnjaka i entuzijasta. 90% ispitanika smatra da postoji manjak svijesti o *chatbotovima* među poduzećima, a inudstrije koje će naviše profitirati su e-trgovina, osiguranja i zdravstvo. 92% ispitanika preferira Facebook Messenger kao mrežu

za implementaciju *chatbota*, a slijede osim osobnih stranica tvrtke, Slack i Twitter što je prikazano Slikom 3.4.



Slika 3.4. *Mreže za implementaciju chatbotova*

IBM Watson je prvi izbor kao platforma za izradu *chatbota* za 61% poduzeća, a slijede wit.ai i Microsoft Bot što je prikazano na Slici 3.5.

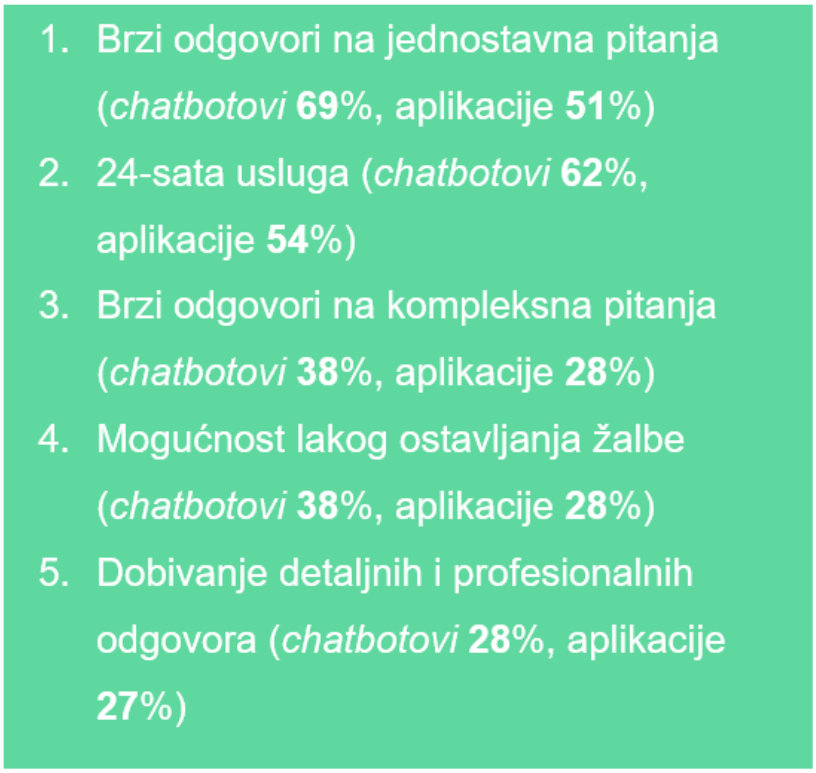


Slika 3.5. *Platforme za izradu chatbotova*

76% ispitanika vjeruje da su resursi za izradu *chatbotova* lako dostupni, a na pitanje „Jesu li *chatbotovi* spremni preuzeti svijet?“ 75% ih misli da se *chatbotovi* nisu još dovoljno iskazali. Prema istraživanju, najpopularniji *chatbotovi* su Poncho, Siri, Exclone, x.ai i Health tap. 67% poduzeća smatra da će *chatbotovi* nadigrati mobilne aplikacije u sljedećih 5 godina. 75% ispitanika planira izgraditi *chatbot* za svoje poduzeće te, od onih koji ih imaju, 95% ih je

zadovoljno načinom na koji im *chatbotovi* pomažu u poslovanju. Ono što ispitanici smatraju da bi se trebalo popraviti kod *chatbotova* su u najvećoj mjeri inteligencija i razgovor. Zaključak istraživanja je da sklonost poduzeća prema povećanju korisničkog iskustva i smanjenu operativnih troškova doprinosi rastu globalnog tržišta *chatbotova*. Također, razvoju *chatbotova* pridonose razvoj umjetne inteligencije, nove platforme za izradu *chatbotova* te dostupnost NLP izvora. Jedino što bi moglo zaustaviti rast *chatbot* tržišta su nedostatak svijesti i velika ovisnost ljudi o interakciji s kupcima kao i pitanja vezana za privatnost.

Novije istraživanje povedeno je 2018. godine od strane pet tvrtki: Drift, SurveyMonkey, Audience, Slaesforce i myclevera ticalo se kako *chatbotovi* mijenjaju način na koji ljudi kupuju i poslovno komuniciraju, a opisano u [19]. Potrošači očekuju brzo i jednostavno pronaći informacije na Internetu te postaju frustrirani kada im poduzeća to ne mogu omogućiti. *Chatbotovi* su spremni ublažiti frustracije pružajući pristup u stvarnom vremenu na zahtjev koji potrošači traže. Tri potencijalne prednosti *chatbotova* koje ispitanici navode su 24-satna usluga, odgovori koji se dobiju istog trenutka te odgovori na jednostavna pitanja. 43% ispitanika preferira stvarnu osobu za komunikaciju radije nego chatbot. U usporedbi s drugim poslovnim komunikacijskim kanalima, *chatbotovi* su postigli drugu najvišu razinu kada su u pitanju potrošači koji očekuju odgovore istog trena, dok se na prvom mjestu našao *online* chat. Korištenjem *chatbota* u kombinaciji s *online chatom*, poduzeća mogu pružiti razinu usluge u stvarnom vremenu. *Chatbotovi* su u pet kategorija prema istraživanju nadjačali mobilne aplikacije što je prikazano Slikom 3.6.



Slika 3.6. *Kategorije u kojima su chatbotovi nadjačali mobilne aplikacije*

3.4. Integracija NLP servisa u chatbot

U kontekstu *chatbota*, integracija NLP-a znači dodavanje ljudske komponente. *Chatbotovi* tada mogu zaista komunicirati s ljudima, a ne samo voditi oskudni razgovor u obliku pitanje-odgovor kako je opisano u [20]. S NLP-em je moguće trenirati *chatbot* na raznim situacijama u kojima se može naći i tako pojednostaviti odgovore koje šalje. U većini slučajeva to treniranje sastoji se od primjera sadržaja s kojima će se *chatbot* susresti. Više primjera tako *chatbotu* daje širu osnovu s kojom može interpretirati pitanja i naredbe i u skladu s time ispravno odgovoriti. Nužnost NLP-a u *chatbotu* ovisi o načinu na koji je *chatbot* izgrađen i što se tim *chatbotom* želi postići. Postoji nekoliko načina kako se to može odrediti. Ako je *chatbot* već izrađen, a postoje podaci o odgovorima s kojima *chatbot* radi, onda je potrebno provjeriti postavljaju li korisnici pitanja i koliko dobro *chatbot* odgovara. Ako je broj pitanja koji korisnici postavljaju mali, NLP u tom slučaju nije potreban. No, ako se *chatbot* suočava s velikim brojem pitanja, a odgovori su mu prilično slabi, NLP može služiti kao nadmoćna metoda za davanje konzistentnih odgovora. Drugo, ako *chatbot* još ne postoji, treba se uvidjeti njegova namjena. Hoće li imati osobnost? Hoće li se uglavnom upravljati gumbima i predloženim odgovorima ili sirovim korisničkim upitima i pitanjima? Ako je cilj imati pričljiv *chatbot* koji radi na načinu da s „razumijevanjem“ odgovara na pitanja koja postavlja korisnik, onda je nužna implementacija NLP-a. Težina implementacije NLP-a ovisi o platformi koja će se koristiti. Nevezano za vrstu *chatbota*, ukoliko se želi integrirati NLP to je potrebno napraviti u četiri faze: pretprocesiranje podataka, izrada modela, treniranje modela i testiranje.

3.4.1. Pretprocesiranje podataka

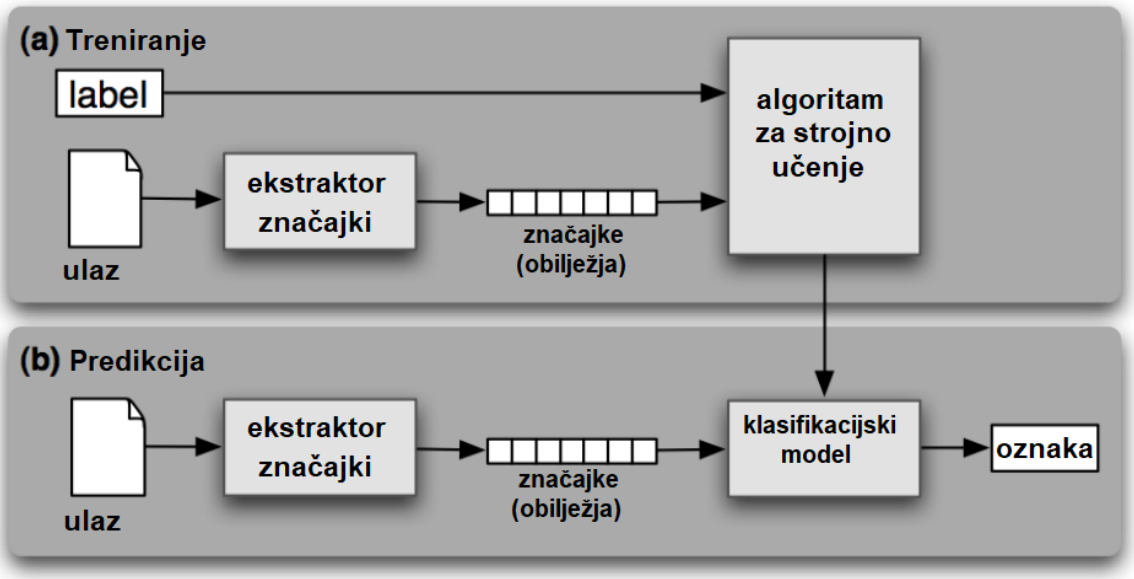
Prema [21] pretprocesiranje podataka tehnika je otkrivanja znanja u skupovima podataka koje uključuje transformiranje sirovih podataka u razumljiv oblik. Podaci su stvarnom svijetu često su nepotpuni, nedosljedni i/ili ih nedostaje u određenim situacijama te je moguće da sadržavaju pogreške. Pretprocesiranje podataka rješava takve probleme pripremajući sirove podatke za daljnju obradu. Tijekom pretprocesiranja podaci prolaze kroz niz koraka: čišćenje, integracija, transformacija, smanjenje i diskretizacija podataka. U čišćenju podataka podaci se čiste kroz procese poput popunjavanja vrijednosti koje nedostaju, zaglađivanja šumova ili rješavanja nedosljednosti u podacima. Kod integracija podataka podaci s različitim prikazima sastavljaju se i tako su riješeni sukobi unutar podataka. Transformacijom se podaci normaliziraju, agregiraju i generaliziraju. Smanjenje podataka za cilj ima prikazati smanjenu zastupljenost podataka u skladištu podataka. Diskretizacija podataka uključuje smanjenje broja vrijednosti kontinuiranog atributa dijeleći raspon atributnih intervala.

3.4.2. Izrada modela

NLP modeli služe za analiziranje korisničkih rečenica upućenih *chatbotu*. Prema [22] razlikuju se modeli temeljeni na pravilima i inteligentni modeli. Modeli temeljeni na pravilima su jednostavni za izradu *chatbotova*, ali je vrlo teško s takvim modelom kreirati *chatbot* koji će odgovarati na kompleksne upite. Podudaranje uzoraka je slabo i *chatbotovi* ne rade dobro kad se susretnu s nepoznatim uzorkom. Također, proces pisanja pravila je dugotrajan i mora se ručno raditi. Ako se želi da *chatbot* uči iz postojećih razgovora, u cijeli proces uključuje se strojno učenje. Takvi modeli su inteligentni, a dijele se na modele temeljene na povratu i generativne modele. Modeli temeljeni na povratu biraju odgovor iz kolekcije odgovora na temelju upita. Ne stvaraju se nove rečenice pa nije potrebna briga za gramatiku. Generativni modeli su prilično inteligentni. Generiraju odgovore riječ po riječ ovisno o upitu. Takvi modeli su skloni gramatičkim pogreškama te su teški za trenirati jer sami strukturiraju rečenice. No, jednom trenirani, nadmašuju modele temeljene na povratu u smislu rukovanja s neviđenim upitima i stvaranje dojma ljudskog razgovora. Jedan takav model je model slijeda sekvenci objašnjen u potpoglavlju 3.2.2.

3.4.3. Treniranje modela

Da bi se model mogao koristiti za predikciju na neviđenim primjerima, prvo se mora trenirati na označenim primjerima [6]. Svaki primjer predstavlja se vektorom značajki *x*. Svaka značajka kodira jedan oblik ulaza, a ekstraktor značajki (engl. *feature extractor*) generira vektor značajki *x* za svaki ulazni primjer. Generiranje vektora značajki isto je i za treniranje i predikciju. Dok se trenira, model na ulazu dobiva primjere *x* i njihove oznake *y*, tj. parove (x, *y*). Tako istreniran model koristi se za predikciju. Kod predikcije model na ulazu dobiva samo primjere *x*, a na izlazu daje oznaku *y*. Opisan postupak grafički je prikazan Slikom 3.7.



Slika 3.7. *Treniranje i predikcija*

3.4.4. Testiranje

Testiranje služi kao provjera točnosti modela. Model je točan ako dobro generalizira, tj. ako dobro predviđa, a da prethodno nije vidio podatke za testiranje. Treniranje modela vrši se na skupu za treniranje. Pomoću treniranog modela seradi predikcija na skupu za testiranje i na kraju se na tom skupu računa točnost. Kod nekih modela moguće je ugađati njihovu složenost, a kako bi se odredila optimalna složenost modela, potreban je treći skup podataka. Treći skup podataka služi za provjeru. Modeli različite složenosti treniraju se na skupu za učenje i ispituju na skupu za provjeru. Nakon odabira optimalnog modela na skupu za provjeru, točnost modela izračunava se na skupu za testiranje.

4. FERITTO CHATBOT

Feritto *chatbot* temeljen je na pravilima, izrađen u Ruby on Rails okruženju te integriran u Facebook Messenger. Komunikacija se ostvaruje preko Facebook Messenger platforme i moguće je dobiti informacije o FERIT-u, kao i odgovore na još neke definirane upite. Za izradu i integraciju *chatbota* u Facebook Messenger najveću ulogu ima 'facebook-messenger' gem koji će se detaljno objasniti, kao i prethodno navedeno Ruby on Rails okruženje. Za integraciju u Facebook Messenger upotrijebljena je Facebook Developers platforma.

4.1. Facebook Messenger

Facebook Messenger ili samo Messenger, platforma je i aplikacija za razmjenu poruka prema [23]. Razvijena je 2008. godine kao „Facebook Chat“ u sklopu Facebook aplikacije, a u kolovozu 2011. godine dobiva zasebne iOS i Android aplikacije. Tijekom godina Facebook je razvio aplikacije za različite operacijske sustave te također izbacio zasebno Messenger *web* sučelje koje ga razdvaja od same Facebook aplikacije. Korisnici preko Messengera mogu slati poruke i izmjenjivati slike, naljepnice, audio i video zapise, datoteke, moguće je uspostaviti video i glasovne pozive, a od travnja 2016. moguće je komunicirati s *chatbotovima*. Facebook omogućuje tvrtkama razvijanje automatizirane korisničke podrške, vodiča za e-kupovinu i interaktivna iskustva preko *chatbota*. Preko novog API-ja Messenger platforme *chatbotovi* su sposobni odgovoriti strukturiranim porukama koje uključuju slike, linkove i gumbe za akciju. Tako je moguće rezervirati mjesto u restoranu, ostaviti recenziju za e-trgovinu i drugo. Platforma trenutno ne omogućuje plaćanje. Razvojni programeri mogu izraditi vlastite *chatbotove* preko Facebook Developers platforme.

4.2. Izrada chatbota

Chatbot je izrađen u Ruby on Rails okruženju uz pomoć 'facebook-messenger' gema. Ruby je objektno orijentirani programski jezik opće namjene. Osmislio ga je i razvio Japanac Yukihiro Matsumoto sredinom 90-tih godina prošlog stoljeća. Prednost Ruby-ja kao programskog jezika jest to što je usmjeren na razvojnog programera. Ovdje je naglasak na tome da program ima logičkog smisla svakome tko ga pročita pa najčešće nije važno što se događa s memorijom i kako se koristi procesor. Ruby on Rails je radni okvir (engl. *framework*) za izradu web aplikacija pisan u programskom jeziku Ruby kako je opisano u [24]. Dizajniran je kako bi učinio programiranje *web* aplikacija što jednostavnijim čineći pretpostavke o tome što je razvojnom programeru potrebno na samom početku. Omogućuje pisanje manje koda, a

izvršavanje više od drugih jezika i radnih okvira. Rails filozofija uključuje dva glavna načela, „ne ponavljaj se“ i „konvencija prije konfiguracije“. Načelo „ne ponavljaj se“ (engl. *Don't Repeat Yourself*, skraćeno DRY) ukazuje na to da se isti kod ne piše na više mjesta projektu čime kod postaje održiv, proširiv i s manje grešaka. Načelo „konvencija prije konfiguracije“ (engl. *Convention Over Configuration*, skraćeno CoC) odnosi se na to da sam Rails ima prethodno postavljena „mišljenja“ kako nešto napraviti u aplikaciji i toga bi se trebalo držati, a ne konfigurirati po vlastitim željama.

RubyGems upravitelj je paketa (engl. *package manager*) za Ruby programski jezik koji pruža standardni format distribucije Ruby programa i biblioteka u samostalnom formatu nazvanom „gem“. U [25] navodi se da je alat dizajniran za jednostavno upravljanje instalacijom gemova i poslužiteljem koji ih je stvorio. Sučelje za RubyGems je alat naredbene linije nazvan gem koji može instalirati i upravljati bibliotekama (gemovima). RubyGems integrira se s Ruby *run-time loaderom* kako bi pomagao instalirati gemove iz standardiziranih mapa biblioteka. Svaki gem sastoji se od tri dijela: kod, dokumentacija i specifikacija.

'facebook-messenger' gem dodaje se u datoteku *Gemfile* i instalira preko terminala naredbom 'gem install facebook-messenger'. Prema službenoj dokumentaciji [26] opisano je da se koristi za slanje i primanje poruka: moguće je odgovoriti na korisnikove poruke, moguće je pisati korisniku ukoliko se primijeti neaktivnost, moguće je slati fotografije te kreirati poruke s brzim odabirom. Također, moguće je slati poruke s gumbima i reagirati na svaki korisnikov odgovor. Moguće je uključiti i isključiti indikator pisanja, označiti poruku pročitanom, snimati poslane poruke i primljene zahtjeve za porukom. Konfiguracija je jednostavna i izvršava se u nekoliko koraka. Prvo je potrebno postaviti *environment* varijable u .env datoteku koje će biti potrebne za kasniju integraciju (ACCESS_TOKEN, APP_SECRET i VERIFY_TOKEN). *Chatbot* će se smjestiti u novu mapu app/bot. U Rails aplikaciji u datoteci config/routes.rb potrebno je usmjeriti server na *chatbot* prema kodu na Slici 4.1.

```
Rails.application.routes.draw do
  mount Facebook::Messenger::Server, at: "bot"
end
```

Slika 4.1. *Usmjeravanje servera na chatbot*

Unutar mape app/bot izrađena je datoteka listen.rb u kojoj se vrši pretplata i nalazi glavni kod aplikacije. Kako bi Rails prepoznao datoteke unutar bot mape, potrebno je dodati kod na Slici 4.2. u config/application.rb datoteku.

```ruby
# Auto-load /bot and its subdirectories
config.paths.add File.join("app", "bot"), glob: File.join("**","*.rb")
config.autoload_paths += Dir[Rails.root.join("app", "bot", "*")]
```

Slika 4.2. *Prepoznavanje datoteka unutar bot mape*

Funkcionalnost *chatbota* sadržana je u listen.rb datoteci. U ovom dijelu vrši se pretplata na Facebook stranicu s Access Tokenom spremljenim u .env datoteku te se *chatbotu* daju funkcionalnosti. Na primljenu poruku u terminalu se ispisuje „Got your message!“, poruka se označava kao pročitana ('message.mark_seen') i vidljiv je indikator pisanja od strane *chatbota* ('message.typing_on') što se vidi u kodu na Slici 4.3.

```ruby
require "facebook/messenger"
include Facebook::Messenger

Facebook::Messenger::Subscriptions.subscribe(access_token: ENV["ACCESS_TOKEN"])

Bot.on :message do |message|
  message.id          # => 'mid.1457764197618:41d102a3e1ae206a38'
  message.sender      # => { 'id' => '1008372609250235' }
  message.seq         # => 73
  message.sent_at     # => 2016-04-22 21:30:36 +0200
  message.text        # => 'Hello, bot!'
  message.attachments # => [ { 'type' => 'image', 'payload' => { 'url' => 'https://
  www.example.com/1.jpg' } } ]

  puts "Got your message!"
  message.mark_seen
  message.typing_on
```

Slika 4.3. *Pretplata*

Feritto je *chatbot* baziran na pravilima što znači da nema integriran NLP, a sva komunikacija oslanja se na prethodno definirana pravila, u ovom slučaju na definirane odgovore. Komunikacija se odvija na engleskom jeziku, a sav kod izveden je kroz if else naredbe. Primjer naredbe u Ruby jeziku nalazi se na Slici 4.4.

```ruby
if message.text.nil?
  message.reply(text: 'Empty message received.')

elsif message.text.include? "hello"
  message.reply(text: 'Hello, human! How can I help you?')

else message.reply(text: "I don't understand you.")

end
```

Slika 4.4. *if-else naredba u Ruby jeziku*

Ovaj dio koda konkretno predstavlja nekoliko uvjeta: ako je poruka prazna, odgovor će biti „Empty message received.“, ako poruka uključuje riječ „hello“ odgovor će biti „Hello, human! How can I help you?“, u svim ostalim slučajevima odgovor će biti „I don't understand you.“ End označava kraj uvjeta.

Sadržaj listen.rb datoteke nalazi se u prilogu P.4.1., a cijeli repozitorij na https://github.com/ljubicab/rubot. Na slici 4.5. može se vidjeti dio funkcionalnosti *chatbota*.

```ruby
require "facebook/messenger"
include Facebook::Messenger

Facebook::Messenger::Subscriptions.subscribe(access_token: ENV["ACCESS_TOKEN"])

Bot.on :message do |message|
  message.id          # => 'mid.1457764197618:41d102a3e1ae206a38'
  message.sender      # => { 'id' => '1008372609250235' }
  message.seq         # => 73
  message.sent_at     # => 2016-04-22 21:30:36 +0200
  message.text        # => 'Hello, bot!'
  message.attachments # => [ { 'type' => 'image', 'payload' => { 'url' => 'https://
  www.example.com/1.jpg' } } ]

  puts "Got your message!"
  message.mark_seen
  message.typing_on

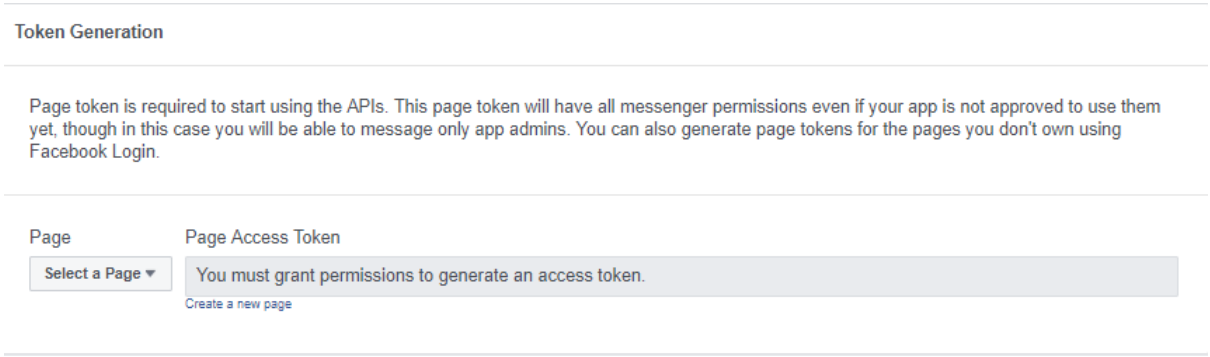
  if message.text.include? "hello" or message.text.include? "hi" or message.text.include? "hey" or
    message.text.include? "Hello" or message.text.include? "Hi" or message.text.include? "Hey"
    message.reply(text: 'Hello, human! How can I help you?')

  elsif message.text.include? "FERIT info"or message.text.include? "information" or message.text.
    include? "give me information"
    message.reply(
      text: 'Choose information',
      quick_replies: [
        {
          content_type: 'text',
          title: 'General',
          payload: 'GENERAL'
        },
        {
          content_type: 'text',
          title: 'Departments',
          payload: 'DEPARTMENTS'
        },
        {
```

Slika 4.5. *Dio funkcionalnosti chatbota*

4.3. Integracija u Facebook Messenger

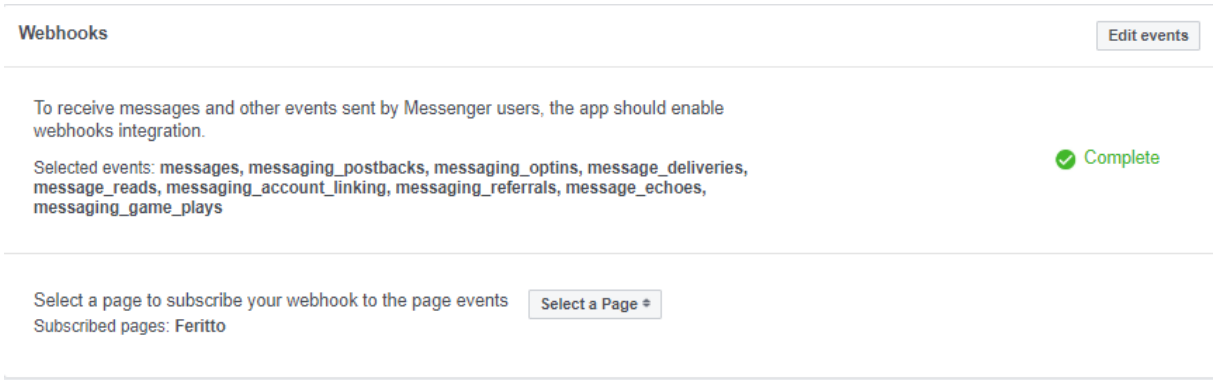
Za integraciju *chatbota* u Facebook Messenger potrebno je izraditi aplikaciju na Facebook Developers Tools platformi. Nakon izrade aplikacije potrebno je izabrati produkte koje će aplikacija sadržavati, u ovom slučaju radi se o Messengeru. Aplikaciju je potrebno povezati s prethodno izrađenom Facebook stranicom, u ovom slučaju Feritto, a stranica je dostupna na https://www.facebook.com/Feritto-315126462586874/. Također, potrebno je uspostaviti *webhook*, tj. utvrditi valjanost URL-a (engl. *Uniform Resource Locator*, prevedeno ujednačeni ili usklađeni lokator sadržaja) do lokalne Rails aplikacije. Kako Rails aplikacija nije javno dostupna, upotrijebljen je besplatni alat ngrok[4] kako bi se stvorio HTTPS tunel do lokalnog *web* servera. Facebook aplikacija se s lokalnom aplikacijom povezuje pomoću Access Tokena, App Secret tokena te Verify Tokena koji je potreban za *webhook*. Access Token generira se na aplikaciji izrađenoj na Facebook Developers platformi vidljivo na Slici 4.4.



Slika 4.4. *Generiranje Access Tokena*

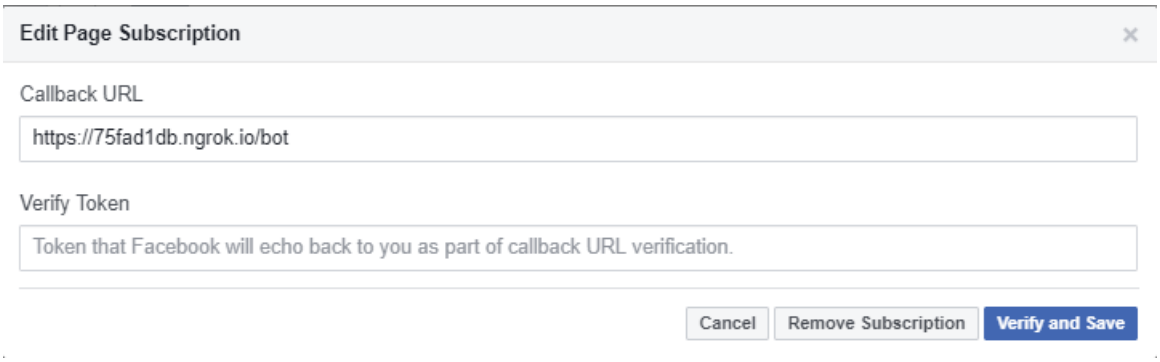
App Secret generira se na dijelu s osnovnim postavkama. Prilikom uspostavljanja *webhooka* bilo je potrebno odabrati događaje na koje će se stranica pretplatiti. Odabrani događaji vidljivi su na Slici 4.5.

[4] Ngrok – secure introspectable tunnels to localhost https://ngrok.com/



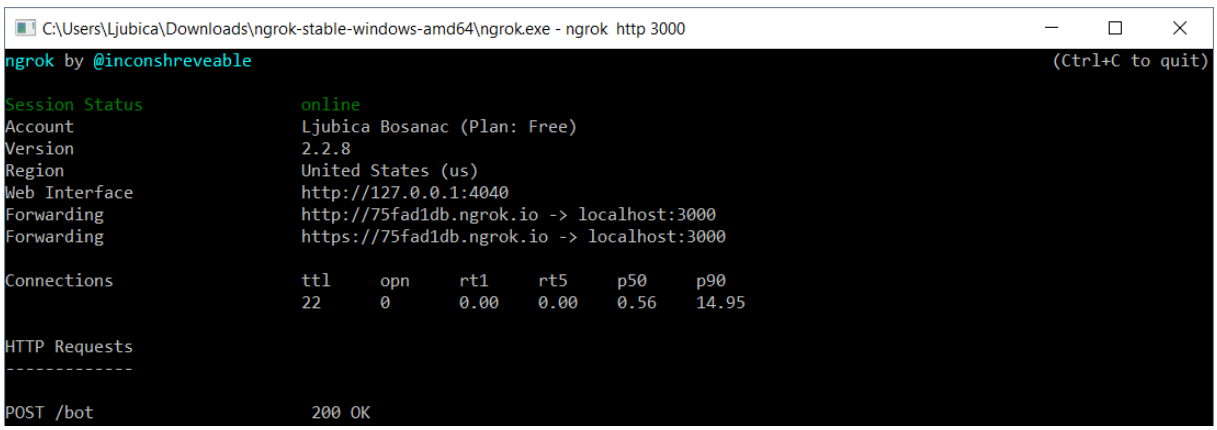
Slika 4.5. *Odabrani događaji na koje je stranica pretplaćena*

Osim događaja, potrebno je upisati povratni URL i Verfiy Token koji je isti Verify Tokenu u Rails aplikaciji. Povratni URL generira se od strane ngrok-a i za svako povezivanje je drukčiji pa je prilikom pokretanja servera svaki put potrebno iznova potvrditi pretplatu. Na Slici 4.6. vidljiva su polja za upis URL-a i Verify Token potrebni za uređivanje pretplate.



Slika 4.6. *Uređivanje pretplate*

Pokretanjem ngrok terminala i upisivanjem naredbe 'ngrok http 3000', gdje 3000 predstavlja port na kome je pokrenut server Rails aplikacije, ostvaruje se tunel. Uspješno ostvarena veza prikazana je na Slici 4.7.



Slika 4.7. *Uspješno ostvarena veza*

Ovaj dio predstavlja zadnji dio integracije *chatbota* u Facebook Messenger, a ispravnost se može provjeriti slanjem poruke na Messengeru imajući u vidu prethodno definirane odgovore u listen.rb datoteci. Za testiranje aplikacije potrebno je imati pokrenut server Rails aplikacije (naredba 'rails s' u terminalu) te uspostavljen *webhook* do lokalnog web servera. Nakon što je to uspostavljeno, u Facebook Messengeru, na desktopu ili mobilnoj aplikaciji, moguće je izmjenjivati poruke s Ferittom. U ovom slučaju testiranje je izvršeno na desktop aplikaciji.

Feritto *chatbot* nije javno dostupan budući da nije prolazio provjeru aplikacije od strane Facebooka. Naime, prema [27] prije nego aplikacija postane javno dostupna, Facebook mora utvrditi da se njihovi podaci i proizvodi koriste na odobren način. Za postizanje toga zahtijevaju da aplikacije prođu provjeru (engl. *App Review*). Općenito, postupak uključuje specificiranje vrste podataka koje će aplikacija tražiti od korisnika i opisivanje kako će se ti podaci koristiti. Ovisno o podnesenom zahtjevu, Facebook može tražiti i dodatne korake. Proces provjere aplikacije sastoji se od sljedećih šest koraka:

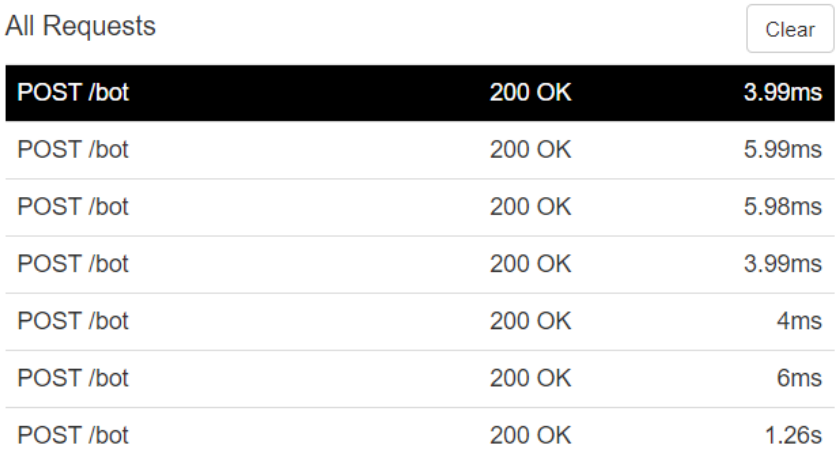
1) Provjera aplikacije
2) Provjera proizvoda
3) Verifikacija tvrtke
4) Dopunski uvjeti
5) Izmjena tehničkog pružatelja usluga
6) Verifikacija krajnjih korisnika tvrtke

Zahtjev za objavom Feritto *chatbot* aplikacije nije podnesen zbog složenosti koraka i količine podataka koja se mora dostaviti. Nadalje, *chatbot* je izrađen u sklopu diplomskog rada i ne postoji namjera za njegovim širim korištenjem.

4.4. Funkcionalnost i testiranje

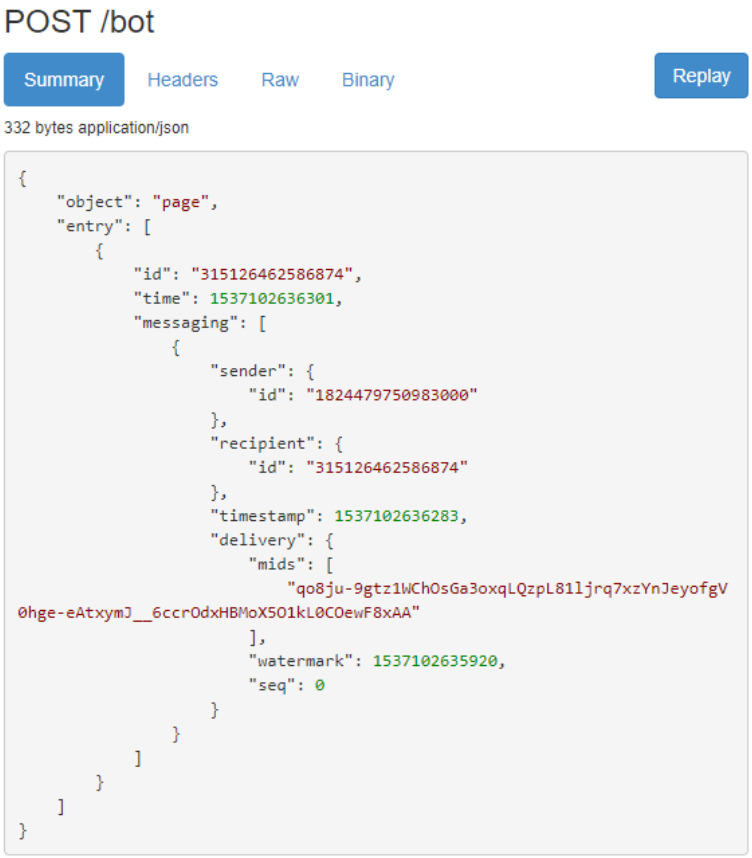
U službenoj dokumentaciji Facebook Messengera, a prema [28] opisano je slanje poruka. Da bi se nekome poslala poruka u Facebook Messengeru, korisnik mora inicijalizirati razgovor. Poruke se klasificiraju u 3 različita tipa. Svaki tip poruke ima različite politike i smjernice za vrste sadržaja i uvjete pod kojima se mogu slati. Postoje standardne, pretplatničke i sponzorirane poruke. Kod standardnih poruka izmjenjuju se obične poruke, a *chatbot* ima 24 sata za odgovor. Pretplatničke poruke omogućuju *chatbotu* slanje poruka korisniku, a sponzorirane poruke su tipovi reklama koje *chatbot* šalje svim otvorenim razgovorima, tj. to su obične poruke u razgovoru koje su označene kao sponzorirane. Feritto *chatbot* koristi standardne poruke. Sve poruke se šalju tako da pošalju POST zahtjev SEND API-ju s Access

Tokenom pridruženom URL upitu. Svi POST zahtjevi vidljivi su u terminalu i na ngrok kontrolnoj ploči koja je dostupna na *localhost* portu 4040. Dio POST zahtjeva vidljiv je na slici 4.8.



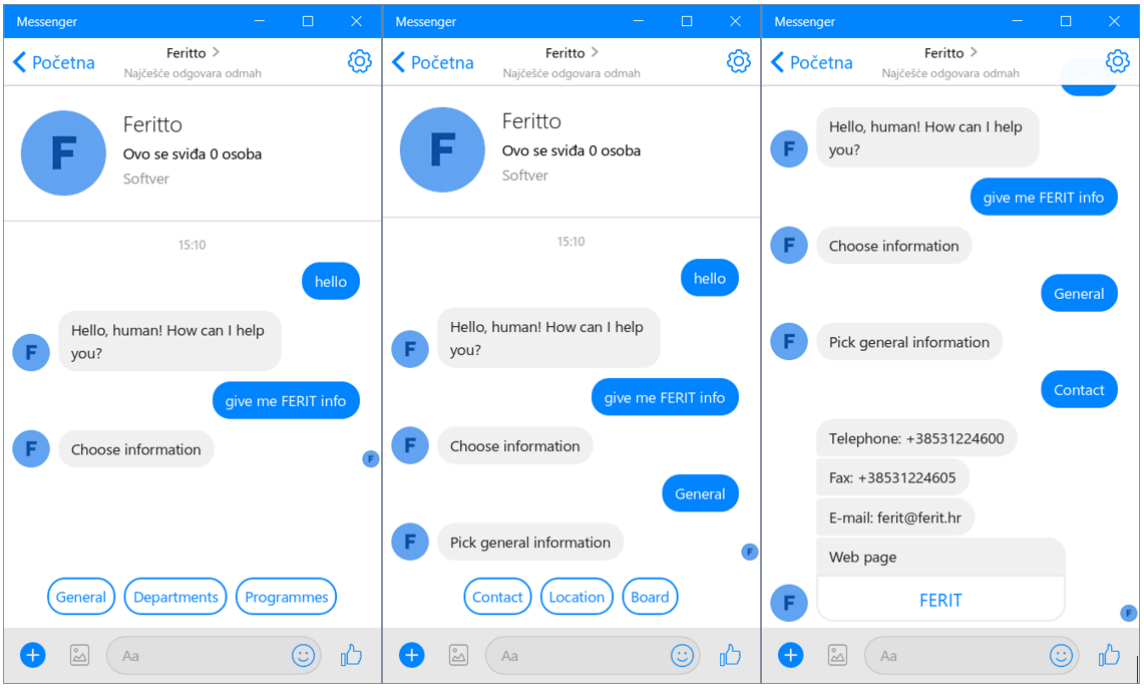
Slika 4.8. *POST zahtjevi*

Svaki takav zahtjev sadržan je u JSON formatu i sadrži podatke o pošiljatelju, primatelju, ID-u poruke, vremenu i dr. što je vidljivo na Slici 4.9.



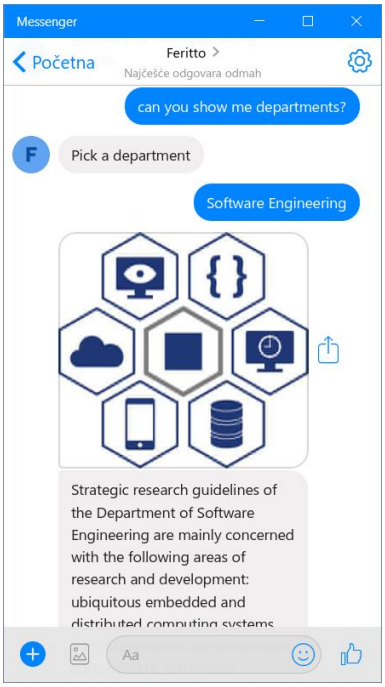
Slika 4.9. *JSON format POST zahtjeva*

Glavna funkcionalnost Feritto *chatbota* je davanje informacija o FERIT-u, ali je programiran da osim službenih informacija može ispričati koju šalu i reagirati na neke korisnikove poruke. Dostupne informacije o FERIT-u podijeljene su na tri kategorije: General, Departments i Programmes, a preuzete sa službene stranice https://www.ferit.unios.hr/. Na odabir „General“ Feritto daje mogućnost odabira općih informacija te ovisno o odabiru šalje odgovor u obliku teksta i/ili priloga. U slučajnu kontakta, osim teksta šalje i prilog u obliku web stranice. Klik na FERIT vodi na web stranicu FERIT-a. Na Slici 4.10. prikazan je razgovor koji vodi do prikazivanja kontakt informacija.



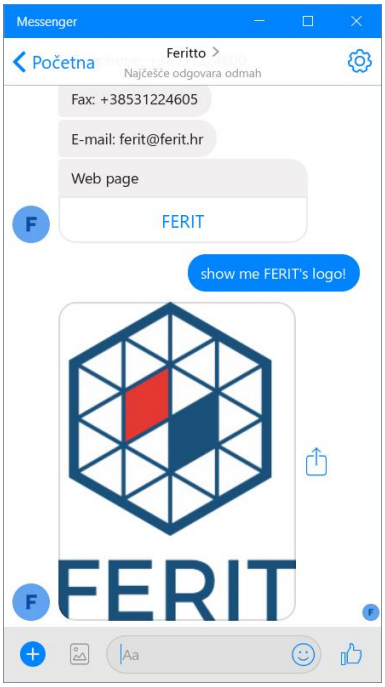
Slika 4.10. *Prikazivanje kontakt informacija*

Osim kontakta, opće informacije nude i lokaciju te odbor. Nadalje, odgovori za zavode (Departments) i programe (Programmes) daju opise ovisno o daljnjem odabiru. Za odabir određenog zavoda Feritto šalje opis zavoda s priloženom slikom preuzete sa stranice FERIT-a vidljivo na Slici 4.11.

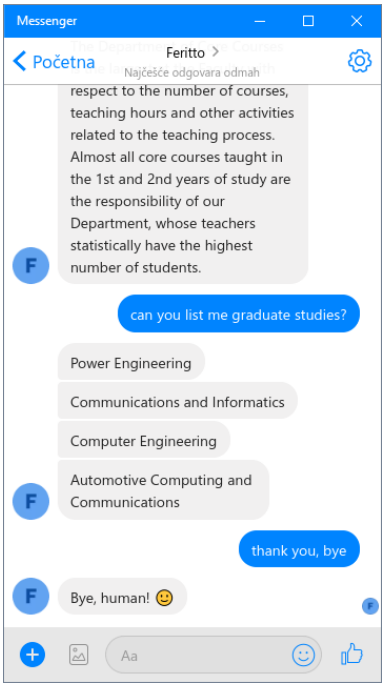


Slika 4.11. *Zavod za programsko inženjerstvo*

Na upit za logo FERIT-a Feritto šalje sliku kao odgovor vidljivo na Slici 4.12., a na Slici 4.13. prikazane su još neke moguće situacije. Tablica s popisom mogućih odgovora nalazi se u prilogu P.4.2.



Slika 4.12. *Logo FERIT-a*



Slika 4.13. *Ispis diplomskih studija i pozdrav*

5. ZAKLJUČAK

Chatbotovi su sustavi koji mogu vršiti razgovore s ciljem oponašanja nestrukturiranih razgovora ili razgovora karakterističnih za međuljudsku interakciju. Za razumijevanje njihovog rada potrebna su znanja o umjetnoj inteligenciji, strojnom učenju, dubokom učenju te generiranju i obradi prirodnog jezika. Umjetna inteligencija znanost je koja omogućuje računalima izvršavanje ljudskih djelatnosti, a počinje se razvijati sredinom prošlog stoljeća. Strojno učenje pristup je koji omogućuje umjetnoj inteligenciji učenje iz iskustva kako bi pronašla uzorke u podacima i znala kako se ponašati u novim situacijama. Obrada prirodnog jezika višedisciplinarno je područje koje se bavi interakcijom ljudskih i računalnih jezika, tj. kako programirati računala za obradu i analizu velikih količina podataka u kojima je sadržan ljudski jezik. Cilj rada bila je izrada i integracija *chatbota* u Facebook Messenger. *Chatbot* Feritto izrađen je u tehnologiji Ruby on Rails te uspješno integriran u Facebook Messenger. Pripada *chatbotovima* baziranima na pravilima, a raspolaže informacijama o FERIT-u. Također, programiran je da odgovori i na neke druge upite. Za integraciju *chatbota* u Facebook Messenger upotrijebljena je Facebook Developers platforma, a zbog složenosti koraka objave aplikacije i količine podataka koja se mora dostaviti Facebooku, *chatbot* je dostupan samo lokalno. Moguće poboljšanje je integracija NLP-a kojom bi se *chatbotu* dodala ljudska komponenta. Tada bi zaista mogao komunicirati s korisnikom, a ne samo voditi razgovor u obliku pitanje-odgovor. Također, javnom objavnom bi se mogla naći primjena u informiranju stranih studenata o fakultetu.

LITERATURA

[1] Umjetna inteligencija: Uvod u umjetnu inteligenciju, izvodi s predavanja, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, akademska godina 2017./2018, http://www2.fer.hr/_download/repository/UI-1-Uvod.pdf (24.6.2018.)

[2] Strojno učenje: Uvod u strojno učenje, izvodi s predavanja, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, akademska godina 2017./2018, https://www.fer.unizg.hr/_download/repository/SU-2017-01-Uvod.pdf (24.6.2018.)

[3] M. M. Mijwl, History of Artificial Intelligence, University of Baghdad, travanj 2015., https://www.researchgate.net/publication/322234922 (23.6.2018.)

[4] Revenues from the artificial intelligence (AI) market worldwide, from 2016 to 2025 (in million U.S. dollars), https://www.statista.com/statistics/607716/worldwide-artificial-intelligence-market-revenues/ (23.6.2018)

[5] European Political Strategy Centre, The Age of Artificial Intelligence: Towards a European Strategy for Human-Centric Machines, izdanje 29, 27. ožujak 2018., https://ec.europa.eu/epsc/sites/epsc/files/epsc_strategicnote_ai.pdf (23.6.2018.)

[6] Umjetna inteligencija: 10. Strojno učenje, izvodi s predavanja, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, akademska godina 2016./2017., https://www.fer.unizg.hr/_download/repository/UI-2017-10-StrojnoUcenje.pdf (24.6.2018.)

[7] Neuronske mreže: Proces učenja, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, https://www.ieee.hr/_download/repository/02-ProcesUcenja-1s.pdf (24.6.2018.)

[8] Umjetne neuronske mreže, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, izvodi s predavanja, 24.5.2011., https://www.fer.unizg.hr/_download/repository/UI_12_UmjetneNeuronskeMreze.pdf (25.6.2018.)

[9] K.S. Jones, Natural Language Processing: a Historical Review, Artificial Intelligence Review (2001): 1-12. listopad 2001. http://www.cl.cam.ac.uk/archive/ksj21/histdw4.pdf (23.6.2018.)

[10] R. Lichtig, The History of Natural Language Processing, San Jose State University, http://ethw.org/The_History_of_Natural_Language_Processing (23.6.2018.)

[11] P. Liang, Natural Language Unterstanding: Foundations and State-of-the-Art, Stanford University, 2017., https://simons.berkeley.edu/sites/default/files/docs/5950/2017.02.01-21.15.12-simons-nlp-tutorial.pdf (22.6.2018.)

[12] H. H. Eckerson, W. W. Eckerson, NaturalLanguage Generation overwie – is NLG worth a thousand pictures?, 2017., https://www.kdnuggets.com/2017/05/nlg-natural-language-generation-overview.html (26.6.2018.)

[13] N. Indurkhya, F. J. Damerau, Handbook of Natural Language Processing, Second Edition, Chapman and Hall/CRC, 2010.

[14] N. Klaić, Š. Anić, Ž. Domović, Rječnik stranih riječi, Zagreb, Sani-plus, 1998.

[15] E. Reiter, R. Dale, Building Natural Language Generation Systems, Cambridge University Press, 2000.

[16] R. Jose, A Study On The Best NLP Tool For Conversational Botmakers, 29.12.2017., https://www.cabotsolutions.com/2017/12/a-study-on-the-best-nlp-tool-for-conversational-botmakers (25.6.2018.)

[17] D. Jurafsky, J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Third Edition draft, 2017., str. 397-402, https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf (22.6.2018.)

[18] Chatbot Survey 2017, Current State of Chatbots and Their Outlook in 2017, https://chatbotsjournal.com/global-chatbot-trends-report-2017-66d2e0ccf3bb (22.6.2018.)

[19] E. Devaney, The 2018 State of Chatbots Report: How Chatbots Are Reshaping Online Experiences, 23.1.2018., https://blog.drift.com/chatbots-report/ (22.6.2018).

[20] J. Crawford, Understanding the Need for NLP in Your Chatbot, 16.2.2018., https://chatbotsmagazine.com/understanding-the-need-for-nlp-in-your-chatbot-78ef2651de84 (26.6.2018.)

[21] Techopedia, What is Data Preprocessing?, https://www.techopedia.com/definition/14650/data-preprocessing (29.6.2018.)

[22] S. Ram, Chatbots with Seq2Seq, 28.6.2016., http://suriyadeepan.github.io/2016-06-28-easy-seq2seq/ (29.6.2018.)

[23] J. Constine, Facebook launches Mesenger platform with chatbots, 12.4.2016., https://techcrunch.com/2016/04/12/agents-on-messenger/?guccounter=1 (12.9.2018.)

[24] Rails Guide, https://guides.rubyonrails.org/getting_started.html (12.9.2018.)

[25] RubyGems, https://www.ruby-lang.org/en/libraries/ (13.9.2018.)

[26] Facebook-messenger gem, https://github.com/jgorset/facebook-messenger (6.9.2018.)

[27] App Review, https://developers.facebook.com/docs/apps/review / (17.9.2018.)

[28] Sending Messages, https://developers.facebook.com/docs/messenger-platform/send-messages (16.9.2018.)

POPIS KRATICA

AI – umjetna inteligencija (engl. *Artificial intelligence*)

AGI – opća umjetna inteligencija (engl. *Artificial General Intelligence*)

ML – strojno učenje (engl. *Machine learning*)

DL – duboko učenje (engl. *Deep learning*)

ANN – umjetna neuronska mreža (engl. *Artificial neural network*)

NLP – obrada prirodnog jezika (engl. *Naural Language Processing*)

NLU – razumijevanje prirodnog jezika (engl. *Natural Language Understanding*)

NLG – generiranje prirodnog jezika (engl. *Natural Language Generation*)

BI – poslovna inteligencija (engl. *Business intelligence*)

IR – pronalaženje informacija (engl. *Information Retrieval*)

AIML – jezik za označavanje umjetne inteligencije (engl. *Artificial Intelligence Markup Language*)

DRY – „ne ponavljaj se“ (engl. *Don't Repeat Yourself*)

CoC – „konvencija prije konfiguracije“ (engl. *Convention over Configuration*)

URL – ujednačeni ili usklađeni lokator sadržaja (engl. Uniform Resource Locator)

SAŽETAK

Cilj rada bila je izrada i integracija *chatbota* u Facebook Messenger. *Chatbot* sustavi ili *chatbotovi* su sustavi koji mogu vršiti razgovore s ciljem oponašanja razgovora karakterističnih za međuljudsku interakciju. Za njihovo razumijevanje potrebno je poznavanje umjetne inteligencije, strojnog učenja, dubokog učenja, obrade prirodnog jezika te razumijevanja i generiranja prirodnog jezika što je detaljno objašnjeno kroz cijeli rad. U tehnologiji Ruby on Rails izrađen je *chatbot* temeljen na pravilima nazvan Feritto. *Chatbot* je integriran u Facebook Messenger te omogućuje komunikaciju s korisnikom. Za integraciju *chatbota* u Facebook Messenger upotrijebljena je Facebook Developers platforma. Feritto raspolaže osnovnim informacijama o FERIT-u te ih ovisno o upitu šalje korisniku. Zbog strogih Facebook pravila *chatbot* nije objavljen i javno dostupan već je uporaba moguća isključivo lokalno. Osim službenih informacija, Feritto je programiran da odgovori i na neke neformalne upite.

Ključne riječi: *chatbot* sustavi, duboko učenje, Facebook Messenger, obrada prirodnog jezika, umjetna inteligencija

ABSTRACT

Chatbot Integration with Facebook Messenger

This paper discusses developing and integrating chatbot with Facebook Messenger. Chatbots are systems that can perform conversations that are similar to human interactions. For their understanding, knowledge of artificial intelligence, machine learning, deep learning, natural language processing, natural language understanding, and natural language generating is needed, which is explained in detail throughout the paper. A rule-based chatbot named Feritto was developed in the Ruby on Rails environment. A chatbot is integrated with Facebook Messenger and allows communication with a user. Facebook Developers platform was used to integrate chatbot with Facebook Messenger. Feritto provides basic information on FERIT and, depending on the query, sends it to the user. Because of the strict Facebook rules, the chatbot is not published and publicly available so its use is only locally enabled. In addition, Feritto has been programmed to also respond to some informal inquiries.

Keywords: artificial intelligence, chatbots, deep learning, Facebook Messenger, machine learning, natural language processing

ŽIVOTOPIS

Ljubica Bosanac rođena je 24.8.1994. u Osijeku. U Valpovu završava opću gimnaziju u sklopu Srednje škole Valpovo. 2013. godine upisuje se na Elektrotehnički fakultet Osijek, danas Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek, te se narednu godinu opredjeljuje za smjer Komunikacije i informatika. Tri semestra bila je demonstrator na kolegijima Fizika 1 i Fizika 2. Na 37. obljetnici Elektrotehničkog fakulteta Osijek nagrađena je priznanjem za uspjeh. U akademskoj godini 2015./2016. polazila je Microsoftovu Software StartUp Akademiju. 2016. godine upisuje Sveučilišni diplomski studij Elektrotehnika, smjer Komunikacije i informatika, izborni blok Mrežne tehnologije. Trenutno radi u BambooLab-u, softverskoj tvrtki za *web* development i digitalni marketing, a hobi joj je sviranje klavira.

PRILOZI

Prilog 1 – sadržaj listen.rb datoteke

```
require "facebook/messenger"
include Facebook::Messenger

Facebook::Messenger::Subscriptions.subscribe(access_token:
ENV["ACCESS_TOKEN"])

Bot.on :message do |message|
  message.id          # => 'mid.1457764197618:41d102a3e1ae206a38'
  message.sender      # => { 'id' => '1008372609250235' }
  message.seq         # => 73
  message.sent_at     # => 2016-04-22 21:30:36 +0200
  message.text        # => 'Hello, bot!'
  message.attachments # => [ { 'type' => 'image', 'payload' => { 'url' =>
'https://www.example.com/1.jpg' } } ]

  puts "Got your message!"
  message.mark_seen
  message.typing_on

  if message.text.nil?
    message.reply(text: 'Empty message received.')

  elsif message.text.include? "hello" or message.text.include? "hi" or
message.text.include? "hey" or message.text.include? "Hello" or
message.text.include? "Hi" or message.text.include? "Hey"
    message.reply(text: 'Hello, human! How can I help you?')

  elsif message.text.include? "FERIT info"or message.text.include?
"information" or message.text.include? "give me information"
    message.reply(
      text: 'Choose information',
      quick_replies: [
        {
          content_type: 'text',
          title: 'General',
          payload: 'GENERAL'
        },
        {
          content_type: 'text',
          title: 'Departments',
          payload: 'DEPARTMENTS'
        },
        {
          content_type: 'text',
          title: 'Programmes',
          payload: 'PROGRAMMES'
        }
      ]
    )

  elsif message.text.include? "General" or message.text.include? "general"
or message.text.include? "general info"
    message.reply(
    text: 'Pick general information',
    quick_replies: [
      {
```

```
        content_type: 'text',
        title: 'Contact',
        payload: 'CONTACT'
      },
      {
        content_type: 'text',
        title: 'Location',
        payload: 'LOCATION'
      },
      {
        content_type: 'text',
        title: 'Board',
        payload: 'BOARD'
      }
    ]
  )

  elsif message.text.include? "Contact" or  message.text.include? "contact"
    message.reply(text: 'Telephone: +38531224600')
    message.reply(text: 'Fax: +38531224605')
    message.reply(text: 'E-mail: ferit@ferit.hr')
    message.reply(
      attachment: {
        type: "template",
        payload: {
          template_type: "button",
          text: "Web page",
          buttons: [
            {
              type: "web_url",
              url: "https://www.ferit.hr",
              title: "FERIT"
            }
          ]
        }
      }
    )

  elsif   message.text.include?   "Telephone"   or   message.text.include?
"telephone"  or   message.text.include?  "phone"  or   message.text.include?
"phone number"
    message.reply(text: 'Telephone: +38531224600')

  elsif  message.text.include?  "Fax"  or  message.text.include?  "fax"  or
message.text.include? "fax number"
    message.reply(text: 'Fax: +38531224605')

  elsif message.text.include? "E-mail" or message.text.include? "Email" or
message.text.include?   "email"   or     message.text.include?   "mail"   or
message.text.include? "mail adress"
    message.reply(text: 'E-mail: ferit@ferit.hr')

  elsif message.text.include? "webpage" or  message.text.include? "web" or
message.text.include? "web place"
    message.reply(
      attachment: {
        type: "template",
        payload: {
          template_type: "button",
          text: "Web page",
          buttons: [
            {
```

```
              type: "web_url",
              url: "https://www.ferit.hr",
              title: "FERIT"
            }
          ]
        }
      }
    )

  elsif message.text.include? "Location" or message.text.include? "location"
or message.text.include? "building" or message.text.include? "Building"
    message.reply(text: 'Address: Kneza Trpimira 2B, HR-31000 Osijek')
    message.reply(text: 'Campus: Cara Hadrijana 10b, HR-31000 Osijek')
    message.reply(
      attachment: {
        type: "template",
        payload: {
          template_type: "button",
          text: "Show on Google Maps",
          buttons: [
            {
              type: "web_url",
              url: "https://goo.gl/maps/iWCUi8UWTfC2",
              title: "Campus"
            },
             {
              type: "web_url",
              url: "https://goo.gl/maps/7ymxZm68VsP2",
              title: "Kneza Trpimira 2B"
            }
          ]
        }
      }
    )

  elsif    message.text.include?    "address"    or    message.text.include?
"trpimirova"  or  message.text.include?  "Address"  or  message.text.include?
"Trpimirova"
    message.reply(text: 'Address: Kneza Trpimira 2B, HR-31000 Osijek')
    message.reply(
      attachment: {
        type: "template",
        payload: {
          template_type: "button",
          text: "Show on Google Maps",
          buttons: [
             {
              type: "web_url",
              url: "https://goo.gl/maps/7ymxZm68VsP2",
              title: "Main address"
            }
          ]
        }
      }
    )

  elsif message.text.include? "campus" or message.text.include? "Campus"
    message.reply(text: 'Campus: Cara Hadrijana 10b, HR-31000 Osijek')
    message.reply(
      attachment: {
        type: "template",
        payload: {
```

```
          template_type: "button",
          text: "Show on Google Maps",
          buttons: [
            {
              type: "web_url",
              url: "https://goo.gl/maps/iWCUi8UWTfC2",
              title: "Campus"
            }
          ]
        }
      }
    )

  elsif message.text.include? "Departments" or message.text.include?
"departments" or message.text.include? "department"
      message.reply(
        text: 'Pick a department',
        quick_replies: [
          {
            content_type: 'text',
            title: 'Core Courses',
            payload: 'CC'
          },
          {
            content_type: 'text',
            title: 'Software Engineering',
            payload: 'SE'
          },
          {
            content_type: 'text',
            title: 'Computer Engineering and Automation',
            payload: 'CEA'
          },
          {
            content_type: 'text',
            title: 'Power Engineering',
            payload: 'PE'
          },
          {
            content_type: 'text',
            title: 'Electromechanical Engineering',
            payload: 'EE'
          },
          {
            content_type: 'text',
            title: 'Communications',
            payload: 'C'
          }
        ]
      )

  elsif message.text.include? "Core Courses" or message.text.include? "core
courses"
    message.reply(
      attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zzp-sve.png'
        }
      }
    )
    message.reply(
```

```
      text: 'The Department of Core Courses is the largest at the Faculty with respect to the number of courses, teaching hours and other activities related to the teaching process. Almost all core courses taught in the 1st and 2nd years of study are the responsibility of our Department, whose teachers statistically have the highest number of students. '
    )

  elsif message.text.include? "Software Engineering" or message.text.include? "software engineering"
    message.reply(
      attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zpi-final.jpg'
        }
      }
    )
    message.reply(
      text: ' Strategic research guidelines of the Department of Software Engineering are mainly concerned with the following areas of research and development: ubiquitous embedded and distributed computing systems based on modern programmable, customisable, distributed computer architectures, and mobile and wireless environments. This includes digital signal processors (DSP), graphics processing units (GPU), as well as cloud and grid computing; algorithms and software solutions for the said platforms with a focus on development of time-critical systems, service-oriented systems, self-sustaining computer systems, big data analytics, which implies the principles of green computing, parallel programming, component-based software engineering, computational intelligence, still image and video processing, and other advanced approaches; procedures and software solutions in medical applications, traffic, mobile and wireless environments, industry and generally in intelligent living spaces.'
    )

  elsif message.text.include? "Computer Engineering and Automation" or message.text.include? "computer engineering and automation"
    message.reply(
      attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zria-sve.png'
        }
      }
    )
    message.reply(
      text: "The Department's mission is to educate students through university and professional study programmes in the fields of automation, robotics and process computing. Moreover, we intend to educate highly professional staff able to follow changes in technology and to apply acquired knowledge to solving engineering problems."
    )

  elsif message.text.include? "Power Engineering" or message.text.include? "power engineering"
    message.reply(
      attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zee-sve.png'
        }
      }
    )
```

```
    message.reply(
      text: 'The Department of Power Engineering is one of the first departments at the Faculty of Electrical Engineering. Rich experience in the field of power engineering has resulted in numerous projects and successfully dealt challenges. '
    )

  elsif message.text.include? "Electromechanical Engineering" or message.text.include? "electromechanical engineering"
    message.reply(
       attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zes-sve.png'
        }
      }
    )
    message.reply(
      text: 'The Department of Electromechanical Engineering is a predecessor of the historical nucleus from which the Faculty of Electrical Engineering Osijek originates, and this is reflected in the fact that activities of this Department are still based upon courses Fundamentals of Electrical Engineering and Electrical Measurements as fundamental courses in the field of electrical engineering.'
    )

  elsif message.text.include? "Communications" or message.text.include? "communications"
    message.reply(
      attachment: {
        type: 'image',
        payload: {
        url: 'https://www.ferit.unios.hr/new-images/zakom-sve.png'
        }
      }
    )
   message.reply(
     text: "The Department's specific goals are in line with the strategic goals of the Faculty of Electrical Engineering, whose final goals include integration into the European Higher Education Area, increase of competitiveness in the global environment with special attention paid to innovations, development of new methods of knowledge and technology transfer to the economy."
    )

  elsif message.text.include? "Board" or message.text.include? "board"
    message.reply(
      text: 'Dean: Dr. Drago Žagar, Full Professor'
    )
    message.reply(
      text: 'Vice-Dean for Education and Student Affairs: Dr. Tomislav Matić, Associate Professor'
      )
    message.reply(
      text: 'Vice-Dean for Science and Postgraduate Studies: Dr. Irena Galić, Associate Professor'
      )
    message.reply(
      text: 'Vice-Dean for Business Activity, Technology and Economic Cooperation: Dr. Kruno Miličević, Associate Professor'
      )
    message.reply(
```

```
      text: 'Vice-Dean for International Cooperation: Dr. Snježana Rimac-
Drlje, Full Professor'
    )

  elsif message.text.include? "Dean" or message.text.include? "dean"
    message.reply(
      text: 'Dean: Dr. Drago Žagar, Full Professor'
    )
    message.reply(
      text: 'Contact: drago.zagar@ferit.hr'
    )

  elsif   message.text.include?   "Programmes"   or   message.text.include?
"programmes"
    message.reply(
      text: 'Pick a programme',
      quick_replies: [
        {
          content_type: 'text',
          title: 'Undergraduate study programme',
          payload: 'UGSP'
        },
        {
          content_type: 'text',
          title: 'Professional study programme',
          payload: 'PSP'
        },
        {
          content_type: 'text',
          title: 'Graduate study programme',
          payload: 'GSP'
        },
        {
          content_type: 'text',
          title: 'Postgraduate doctoral study programme',
          payload: 'PDSP'
        },
        {
          content_type: 'text',
          title: 'Postgraduate specialist study programme',
          payload: 'PSSP'
        }
      ]
    )

  elsif message.text.include? "Undergraduate study" or message.text.include?
"undergraduate study" or message.text.include? "undergraduate studies" or
message.text.include? "Undergraduate studies"
    message.reply(
      text: "Electrical Engineering"
    )
    message.reply(
      text: "Computer Engineering"
    )

  elsif message.text.include? "Professional study" or message.text.include?
"Professional  studies"or  message.text.include?  "professional  study"  or
message.text.include? "professional studies"
    message.reply(
      text: "Automation"
    )
     message.reply(
```

```
      text: "Informatics"
    )
     message.reply(
      text: "Power Engineering"
    )

  elsif message.text.include? "Graduate study" or message.text.include?
"graduate study programme" or message.text.include? "graduate studies" or
message.text.include? "Graduate studies"
    message.reply(
      text: "Power Engineering"
    )
    message.reply(
      text: "Communications and Informatics"
    )
      message.reply(
      text: " Computer Engineering"
    )
    message.reply(
      text: "Automotive Computing and Communications"
    )

  elsif      message.text.include?      "Postgraduate      doctoral"      or
message.text.include? "postgraduate doctoral"
    message.reply(
      text:"Module Power Engineering"
    )
    message.reply(
      text:"Module Communications and Informatics"
    )
    message.reply(
      text:"Module Computer Science "
    )

  elsif     message.text.include?     "Postgraduate     specialist"     or
message.text.include? "postgraduate specialist"
    message.reply(
      text: "Electric Power Networks in the Market Environment"
    )
    message.reply(
      text: "Advanced Communication Technologies"
    )
    message.reply(
      text: "Process Computing"
    )

  elsif message.text.include? "Logo" or message.text.include? "logo"
    message.reply(
      attachment: {
      type: 'image',
      payload: {
        url:                          'http://sociallab.fer.hr/wordpress/wp-
content/uploads/2017/10/ferit-logo-znak-223x300-223x300.png'
        }
      }
    )

  elsif message.text.include? "Talk to me" or message.text.include? "talk to
me" or message.text.include? "let's talk" or message.text.include? "chat" or
message.text.include? "Chat"
    message.reply(text: 'What do you want to talk about?')
```

```
  elsif message.text.include? "weather" or message.text.include? "Weather"
    message.reply(text: "It's a beautiful day, isn't it.")

  elsif message.text.include? "movies" or message.text.include? "Movies"
    message.reply(
      text: "What's your favorite genre?",
      quick_replies: [
        {
          content_type: 'text',
          title: 'Action',
          payload: 'ACTION'
        },
        {
          content_type: 'text',
          title: 'Drama',
          payload: 'DRAMA'
        },
        {
          content_type: 'text',
          title: 'Romance',
          payload: 'ROMANCE'
        }
      ]
    )

  elsif message.text.include? "Yours?" or message.text.include? "yours?"
    message.reply(text: 'Same!')

  elsif message.text.include? "your name"
    message.reply(text: 'My name is Ferrito')

  elsif message.text.include? "tell me a joke" or message.text.include? "tell
me a joke"
    message.reply(text: "Which country's capital has the fastest-growing
population?")
    message.reply(text: "Ireland. Every day it's Dublin.")

  elsif message.text.include? "another joke"
    message.reply(text: "What do you call a bee that can't make up its mind?")
    message.reply(text: "A maybe.")

  elsif message.text.include? "who are you" or message.text.include? "Who are
you"
    message.reply(text: 'Chatbot of FERIT :)')

  elsif message.text.include? "What are you" or message.text.include? "what
are you"
    message.reply(text: 'Just some random chatbot.')

  elsif message.text.include? "Ferrito" or message.text.include? "ferrito"
    message.reply(text: "It's Feritto!")

  elsif message.text.include? "Bye" or message.text.include? "bye"
    message.reply(text: "Bye, human! :)")

  else
     message.reply(text: "Sorry, I don't understand you.")

  end

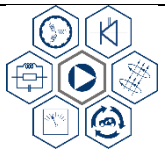
end
```

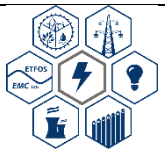
Prilog 2 – tablica mogućih odgovora

U tablici 4.1. ispisane su programirane kombinacije. Korisnički upit predstavlja rečenicu koja sadrži jednu od navedenih riječi

Tablica 4.1.

Korisnički upit	Odgovor chatbota
Hey, Hello, Hi, hey, hello, hi	Hello human! How can I help you?
FERIT info, information, give me information	Choose information General/Departments/Programmes
General, general, general info	Pick general information Contact/Location/Board
Contact, contact	Telephone: +38531224600 Fax: +38531224605 E-mail: ferit@ferit.hr Web page https://www.ferit.hr
Telephone, telephone, phone, phone number	Telephone: +38531224600
Fax, fax, fax number	Fax: +38531224605
E-mail, Email, email, mail, mail address	E-mail: ferit@ferit.hr
Location, location, building, Building	Address: Kneza Trpimira 2B, HR-31000 Osijek Campus: Cara Hadrijana 10b, HR-31000 Osijek Show on Google Maps https://goo.gl/maps/iWCUi8UWTfC2" https://goo.gl/maps/7ymxZm68VsP2
Address, Trpimirova, address, trpimirova	Address: Kneza Trpimira 2B, HR-31000 Osijek Show on Google Maps https://goo.gl/maps/7ymxZm68VsP2
Campus, campus	Campus: Cara Hadrijana 10b, HR-31000 Osijek Show on Google Maps https://goo.gl/maps/iWCUi8UWTfC2
Departments, departments, department	Core Courses, Software Engineering, Computer Engineering and Automation, Power Engineering, Electromechanical Engineering, Communications
Core courses, core courses	 The Department of Core Courses is the largest at the Faculty with respect to the number of courses, teaching hours and

	other activities related to the teaching process. Almost all core courses taught in the 1st and 2nd years of study are the responsibility of our Department, whose teachers statistically have the highest number of students.
Software Engineering, software engineering	 Strategic research guidelines of the Department of Software Engineering are mainly concerned with the following areas of research and development: ubiquitous embedded and distributed computing systems based on modern programmable, customisable, distributed computer architectures, and mobile and wireless environments. This includes digital signal processors (DSP), graphics processing units (GPU), as well as cloud and grid computing; algorithms and software solutions for the said platforms with a focus on development of time-critical systems, service-oriented systems, self-sustaining computer systems, big data analytics, which implies the principles of green computing, parallel programming, component-based software engineering, computational intelligence, still image and video processing, and other advanced approaches; procedures and software solutions in medical applications, traffic, mobile and wireless environments, industry and generally in intelligent living spaces.
Computer Engineering and Automation, computer engineering and automation	 The Department's mission is to educate students through university and professional study programmes in the fields of automation, robotics and process computing. Moreover, we intend to educate highly professional staff able to follow changes in technology and to apply acquired knowledge to solving engineering problems.
Electromechanical Engineering, electromechanical engineering	 The Department of Electromechanical Engineering is a predecessor of the historical nucleus from which the Faculty of Electrical Engineering Osijek originates, and this is reflected in the fact that activities of this Department are still

	based upon courses Fundamentals of Electrical Engineering and Electrical Measurements as fundamental courses in the field of electrical engineering.
Power Engineering, power engineering	 The Department of Electromechanical Engineering is a predecessor of the historical nucleus from which the Faculty of Electrical Engineering Osijek originates, and this is reflected in the fact that activities of this Department are still based upon courses Fundamentals of Electrical Engineering and Electrical Measurements as fundamental courses in the field of electrical engineering.
Communications, communications	 The Department's specific goals are in line with the strategic goals of the Faculty of Electrical Engineering, whose final goals include integration into the European Higher Education Area, increase of competitiveness in the global environment with special attention paid to innovations, development of new methods of knowledge and technology transfer to the economy.
Bord, board	Dean: Dr. Drago Žagar, Full Professor Vice-Dean for Education and Student Affairs: Dr. Tomislav Matić, Associate Professor Vice-Dean for Science and Postgraduate Studies: Dr. Irena Galić, Associate Professor Vice-Dean for Business Activity, Technology and Economic Cooperation: Dr. Kruno Miličević, Associate Professor Vice-Dean for International Cooperation: Dr. Snježana Rimac-Drlje, Full Professor
Dean, dean	Dean: Dr. Drago Žagar, Full Professor Contact: drago.zagar@ferit.hr
Programmes, programmes	Pick a programme Undergraduate study programme/ Professional study programme/ Graduate study programme/ Postgraduate doctoral study programme/ Postgraduate specialist study programme

Undergraduate study, Undergraduate studies, undergraduate study, undergraduate studies	Electrical Engineering Computer Engineering
Professional study, Professional studies, professional study, professional studies	Automation Informatics Power Engineering
Graduate study, Graduate studies, graduate study, graduate studies	Power Engineering Communications and Informatics Computer Engineering Automotive Computing and Communications
Postgraduate doctoral, postgraduate doctoral	Module Power Engineering Module Communications and Informatics Module Computer Science
Postgraduate specialist, postgraduate specialist	Electric Power Networks in the Market Environment Advanced Communication Technologies Process Computing
Logo, logo	
Talk to me, talk to me, let's talk, chat, Chat	What do you want to talk about?
Weather, weather	It's a beautiful day, isn't it.
Movies, movies	Action Drama Romance
Yours?, yours?	Same!
Tell me a joke, tell me a joke	Which country's capital has the fastest-growing population? Ireland. Every day it's Dublin.
another joke	What do you call a bee that can't make up its mind? A maybe.
Who are you, who are you	Chatbot of FERIT :)
What are you, what are you	Just some random chatbot.
Ferrito, ferrito	It's Feritto!
Bye, bye	Bye, human! :)
	Empty message received.
Sve ostalo	Sorry, I don’t understand you.