ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

შალვა ჯაშიაშვილი

კონვოლუციური ნეირონული ქსელების გამოყენება ტექსტების კლასიფიკაციისთვის

ინფორმაციული სისტემები

სამაგისტრო ნაშრომი შესრულებულია ინფორმაციული სისტემების მაგისტრის აკადემიური ხარისხის მოსაპოვებლად

სამაგისტრო ნაშრომის ხელმძღვანელი: გელა ბესიაშვილი

(ტექნიკურ მეცნიერებათა კანდიდატი, ასისტენტ პროფესორი)

თბილისი, 2016

სარჩევი

ანოტაცია

თანამედროვე ტექნოლოგიების გამოყენებით, ინფორმაცია იმდენად სწრაფად ვრცელდება, რომ საჭირო ხდება მისი დახარისხების და დამუშავების მეთოდების მოძებნა, რათა ადამიანს გაუადვილდეს ამ ინფორმაციის გააზრება და შემდგომში გამოყენება.

ვინაიდან ინფორმაციის შენახვისა და გავრცელების ერთ-ერთი ყველაზე ფართოდ გამოყენებადი ფორმა ტექსტური ფორმატია, აქტუალურ ხდება ტექსტების კლასიფიკაციის პრობლემა. ტექსტების კლასიფიკაციის კონკრეტული მაგალითია სპამის ფილტრაცია. გარდა სპამისა, ტექსტების კლასიფიკაციას შეიძლება ჰქონდეს უამრავი გამოყენება, მაგალითად: ონლაინ მაღაზიაში კომენტარების მიხედვით პროდუქტის ხარისხის დადგენა.

ტექსტების კლასიფიკაცია სხვა ტიპის ინფორმაციის კლასიფიკაციისგან განსხვავდება იმით, რომ თითოეულ სიტყვას, სხვადასხვა კონტექსტში განსხვავებული მნიშვნელობა აქვს, რაც ართულებს ამოცანას.

ნაშრომის მიზანია შეიქმნას სისტემა, რომლის გამოყენებითაც მომხმარებელი ააგებს მისთვის სასურველ მოდელს და ამ მოდელის გამოყენებით დაადგენს შეტანილი ტექსტური ინფორმაციის სენტიმენტს (დადებითია შეფასება თუ უარყოფითი).
ამოცანაში მოდელის ასაგებად გამოყენებულია ფილმებზე კომენტარების ბაზა.
აღნიშნული მოდელის მეშვეობით შეგვიძლია 97% სიზუსტით დავადგინოთ დადებითია ფილმის შეფასება თუ უარყოფითი.
ამ ამოცანის გადასაჭრელად გამოყენებულია ნეირონული ქსელის კერძო ტიპი, კონვოლუციური ნეირონული ქსელი, რომელიც აქამდე ძირითადად გამოიყენებოდა მულტიმედიური ტიპის ინფორმაციის კლასიფიკაციისთვის, თუმცა ამ კვლევებმა გვიჩვენა, რომ კონვოლუციები წარმატებით შეგვიძლია გამოვიყენოთ ტექსტების კლასიფიკაციისთვისაც. კონვოლუციები საშუალებას გვაძლევს დასწავლის პროცესში გავითვალისწინოთ სიტყვების კონტექსტი, რაც ჭრის ტექსტის კლასიფიკაციის მთავარ პრობლემას, კონტექსტის დაკარგვას და ამასთანავე აუმჯობესებს მიღებულ შედეგებს.

Abstract

Modern technology enables information to spread so quickly that it becomes necessary to invent new methods of sorting and processing, thus making it easier to analyze the information and put it to use.

One of the most widely used forms of information storage and distribution is textual format, which makes text classification problem rather acute. A specific example of text classification is spam filtering. Apart from this, classification of the texts may have other uses, for example: determining product quality according to comments in the online stores.

Text classification is different from other types of classification of information, since each word may have a different meaning in different contexts, which complicates the task.
The goal of this thesis is to create a system which allows users to create a desired model. The model that can determine the sentiments of textual information (rate is positive or negative).

To construct this model a database of films' comments was used. The model can estimate with 97% accuracy whether a comment which rates a film is positive or negative.

To solve this problem a specific type of neural network is used, a convolutional neural network. This type of neural network was mainly used to classify multimedia type of information. However, resent research shows that the convolution can be successfully used to classify textual data. Convolution allows us to take into account the context in which the word is used, thus solving the main problem of text classification, loss of context, and improving obtained results.

შესავალი

როგორც აღვნიშნეთ, თანამედროვე ტექნოლოგიებმა დააჩქარა ინფორმაციის გავრცელება, ამას ხელი შეუწყო სოციალური ქსელების შექმნამ. ყოველდღიურად ინტერნეტში ჩნდება მილიარდობით სტრიქონი ტექსტური ინფორმაცია. ცხადია ეს ინფორმაცია შეიძლება გამოყენებული იქნას სხვადასხვა მომსახურების გაუმჯობესებისა და დახვეწის, აგრეთვე ახალი მომსახურების შექმნის მიზნით, მაგრამ ინფორმაციის მოცულობიდან გამომდინარე, მის დასამუშავებლად ადამიანური რესურსი საკმარისი არაა, ამიტომ საჭირო გახდა კომპიუტერული რესურსის გამოყენება. პრობლემა მდგომარეობს იმაში, რომ კომპიუტერს არ ესმის ადამიანური დამწერლობიდან არცერთი, ამიტომ საჭირო გახდა ისეთი ალგორითმების შექმნა, რომლის დახმარებითაც კომპიუტერი გაიგებდა ადამიანურ ენას.

კომპიუტერის მიერ, ტექსტური მონაცემების დამუშავების მთავარ ამოცანას წარმოადგენს ტექსტების კლასიფიკაცია. 21-ე საუკუნუს დასაწყისიდან ამ ამოცანის გადასაჭრელად გამოიყენებოდა სხვადასხვა მოდელები: “Bag of words feature extraction”, “Naive Bayes classifier”, “decision tree classifier” და სხვა. თუმცა არცერთ მოდელს არ ჰქონია ისეთი შთამბეჭდავი შედეგი, როგორიც ჰქონდა კონვოლუციური ნეირონული ქსელების გამოყენებას. ბოლო წლებამდე კონვოლუციურ ნეირონულ ქსელებს იყენებდნენ მულტიმედიური ინფორმაციის კლასიფიკაციისათვის, მაგალითად: სახის ამოცნობა, სიმბოლოების ამოცნობა და ა.შ. უკანასკნელმა კვლევებმა აჩვენა, რომ აღნიშნული მოდელი წარმატებით შეგვიძლია გამოვიყენოთ ტექსტური ინფორმაციის დამუშავებაშიც. მარტინ კარპატის მიერ შექმნილი წინადადებების გენერატორი, რომლის ნეირონულმა ქსელმაც შეისწავლა შექსპირის პიესები, “საუბრობს” შექსპირის ტერმინებით. კომპანია IBM-ის მიერ შექმნილ ხელოვნურ ინტელექტში “IBM Watson”, ჩადებულია აღნიშნული მოდელის IBM-ის მოდიფიკაცია, რომელიც ჯერჯერობით არაა ხელმისაწვდომი. აღნიშნულს სისტემას დამუშავებული აქვს ამერიკის კონსტიტუცია და მილიონობით სასამართლო გადაწყვეტილება, ამ ინფორმაციაზე დაყრდნობით მას შეუძლია იყოს უკეთესი ადვოკატი, ვიდრე ადამიანი.

მანქანური სწავლება

რა არის მანქანური სწავლება?

ML არ არის ახალი მეცნიერება. ის ვითარდებოდა როგორც ხელოვნური ინტელექტის ნაწილი 1990 წლამდე. გამოყოფის შემდეგ მეცნიერების ეს დარგი გადაერთო პრაგმატული ამოცანების გადაჭრაზე. ML-ისადმი ინტერესი ბოლო დროს გაიზარდა, რასაც თავის მიზეზები აქვს. გაუმჯობესდა კომპიუტერის მონაცემები, გაიფდა მეხსიერება, აგრეთვე გაიზარდა ხელმისაწვდომი მონაცემების რაოდენობა. რაც თავის მხრივ ნიშნავს რომ შესაძლებელი გახდა უფრო დიდი ინფორმაციის უფრო სწრაფი და ხარისხიანი დამუშავება. ეს კი საშუალებას აძლევს კომპიუტერს უკეთესი გადაწყვეტილებები მიიღოს ადმანიანის ჩარევის გარეშე.

1959 წელს არტურ სამუელმა განსაზღვრა მანქანური სწავლება :„Field of study that gives computers the ability to learn without being explicitly programmed “(მეცნიერების დარგი რომელიც აძლევს კომპიუტერებს სასუალებას დაისწავლოს წინასწარი პროგრამირების გარეშე).

მანქანური სწავლება შეისწავლის ალგორითმებს, რომლებიც საშუალებას აძლევენ კომპიუტერს დაისწავლონ ამა თუ იმ საქმის კეთება. მაგალითად : დაისწავლოს კონკრეტული დავალების შესრულება, ჩამოაყალიბოს ზუსტი ვარაუდები ან მიიღოს გადაწყვეტილებები დამოუკიდებლად. დასწავლო ყველთვის ხდება რაიმე კონკრეტული მონაცემების, ინფრომაციის, გამოცდილებისა და მითითებების ანალიზის ხარჯზე. მანქანური სწავლება არის ხელოვნური ინტელექტის ქვესფერო. მისი ძირითადი მიზანია ისეთი ალგორითმების შემუშავება რომლებიც დამოუკიდებლად შეძლებენ დასწავლას. აქედან გამომდინარე ამ ქვედარგს საკმაოდ დიდი ადგილი უჭირავს ხელოვნური ინტელექტის სფეროში, ვინაიდან ძნელია წარმოვიდგინოთ „ინტელექტი“ რომელსაც დასწავლის უნარი არ გააჩნია.

მანქანური სწავლება ასევე მჭიდროდ ურთიერთქმედებს მეცნიერების სხვა დარგებთან როგორიცაა სტატისტიკა,ფიზიკა, მათემატიკის სხვა დარგები და ა.შ.

მანქანური სწავლების ძირითადი მიზანი არის ზოგადი გამოყენების ალგორითმების შექმნა, რომლის გამოყენებაც უნდა შეგვეძლოს განსხვავებული პრობლემების გადაჭრისას.

მანქანურ სწავლებაში გამოყენებული მეთოდები

მანქანურ სწავლებაში გამოყოფენ სამ ძირითად მეთოდს:

1. კონტროლირებადი დასწავლა(Supervised learning)
2. არაკონტროლირებადი დასწავლა(Unsupervised learning)
3. Reinforcement learning

კონტროლირებადი დასწავლა(Supervised learning)

კონტროლირებადი სწავლების მიზანია ააგოს მოდელი რომლიც დასწავლის მერე მოგვცემს სავარაუდო პასუხს ახალ(არა დასწავლულ) ინფორმაციაზე. ამ მეთოდის დროს, კომპიუტერი სწავლობს მიწოდებულ ინფორმაციაში ადაპტირებადი ალგორითმების დახმარებით, კანონზომიერებების ამოცნობის გზით. როდესაც ამ მოდელს მიეწოდება ახალი ინფორმაცია ის აუმჯობესებს სავარაუდო პასუხის სიზუსტეს.

უფრო კონკრეტულად რომ ვთქვათ კონტროლირებადი სწავლების ალგორითმები იღებენ პარამეტრად ნაცნობ ინფორმაციას(ებს) და ამ ინფორმაციის შესაბამის სასურველ პასუხს(ებს), შემდეგ მიღებული პარამეტერების საფუძველზე „ასწავლიან“ მოდელს რათა ამ უკანასკნელმა შემდგომში უცნობი ინფორმაციის საპასუხოდ აკურატული(რეალურთან მაქსიმალურად ახლო) სავარაუდო რესულტატი დაგვიბრუნოს.

მაგალითი:

ვთქვათ გვინდა გავარკვიოთ მოუვა თუ არა პაციენტს გულის შეტევა. გვაქვს საჭირო ინფორმაცია(ასაკი,წონა,სიმაღლე,სისხლის წნევა და ა.შ) ძველი პაციენტების შესახებ, ასევე ვიცით მოუვიდათ თუ არა მათ გულის შეტევა ამ ინფორმაციის აღებიდან ერთი წლის მერე. ჩვენი მიზანია ამ ინფორმაციის საფუძველზე შევადგინოთ მოდელი რომელიც დასწავლის შემდეგ დაგვიბრუნებს სავარაუდო პასუხს ჩვენთვის საინტერესო კითხვაზე. შესატანი ინფორმაცია წარმოვიდგინოთ როგორც არაერთგვაროვანი მატრიცა სადაც სტრიქონები იქნება მაგალითები, ხოლო სვეტები ინფრომაცია პაციენტზე(ასაკი,წონა,სიმაღლე,სისხლის წნევა და ა.შ). დაბრუნებული პასუხი წარმოვიდგინოთ როგორც სვეტური ვექტორი სადაც თითოეული ელემენტი იქნება მატრიცის შესაბამისი სვეტის მაგალითის საფუძველზე დაგენერირებული სავარაუდო რესულტატი(აქვს თუ არა პაციენტს კიბო). ახლა კი იმისთვის რომ მოდელს „ვასწავლოთ“ საჭიროა ავარჩიოთ შესაბამისი ალგორითმი, რომელსაც გადავცემთ შესატან

ინფორმაციასა და მიღებულ პასუხებს რის შედეგადაც ეს ალგორითმი ჩვენს მოდელს „შეცვლის“,, „მოარგებს“ რეალურ რეზულტატებს.

ამის შემდეგ უკვე შეგვიძლია გამოვიყენოთ მიღებული მოდელი ახალ პაციენტზე რათა გავარკვიოთ ექნება თუ არა მას გულის შეტევა ერთი წლის განმავლობაში.

კონტროლირებადი სწავლება იყოფა ორ დიდ კლასად:

1. კლასიფიკაცია: ამ ჯგუფის მიზანია მის მიერ გამოკვლევად ინფორმაციას მიამაგროს კლასი/იარლიყი კლასების/იარლიყების სასრული რაოდენობის მქონე სიმრავლიდან. ამ ჯგუფის ამოცანებს მიეკუთვნება სპამის ამოცნობა, რეკლამების შემოთავაზების სისტემა, სურათების და საუბრის ამოცნობა და ა.შ. ამ ჯგუფში შედის ზემოთ განხილული მაგალითიც
2. რეგრესია: ამ ჯგუფის მიზანია სავარაუდო პასუხად დაგვიბრუნოს უწყვეტი მაჩვენებელი მის მიერ დაკვირვებად ინფორმაციაზე. მაგალითად რა იქნება აქციების კურსი, როგორი იქნება ენერგიის მოხმარება, ან რამდენი ადამიანი დაავადდება ამა თუ იმ დაავადებით.

კონტროლირებადი დასწავლა უმეტეს შემთხვევაში შეგვიძლია ავღწეროთ შემდეგი ბიჯებით:

1. ინფორმაციის მომზადება
2. ალგორითმის ამორჩევა
3. მოდელის მორგება
4. შემოწმების მეთოდის ამორჩევა
5. მორგების მერე რესულტატის შემოწმება და 3 ბიჯის მანამ გამეორება, სანამ მიღებული რეზუტატი არ დაგვაკმაყოფილებს
6. მიღებული მოდელის გამოყენება სავარაუდო რესულტატის მისაღებად

არაკონტროლირებადი დასწავლა(Unsupervised learning)

გაომიყენება ისეთი მონაციმეებისთვის რომელსაც სასურველი შედეგები არ აქვს. ალგორითმი თვითონ უნდა მიხვდეს რა ინფორმაციასთან აქვს კავშირი. მისი მიზანია გამოიკვლიოს მონაცემები და აღმოაჩინოს მასში რაიმე სახის სტრუქტურა. ამ ტიპის სწავლება კარგად გამოიყენება Transaction data - ზე . მაგაილთად, მოძებნოს მომხმარებლების სეგმენტი მსგავსი თვისებებით რაც საშუალებას მოგვცემს მსგავს სეგმენტებში შემავალ პიროვნებებს მსგავსად მოვექცეთ მარკეტინგული კომპანიისას ხშირად გამოყენებადი კლასტერიზაციის ალგორითმებია:

- Hierarchical clustering
- k-Means clustering
- Gaussian mixture models
- Self-organizing maps
- Hidden Markov models

არაკონტროლირებადი დასწავლის მეთოდი გამოიყენება ბიოინფორმატიკაში გენების კლასტერიზაციისა და სიმრავლეში თანმიმდევრობის ანალიზისთვის, data mining -ში მიმდევრობის და კანონზომიერებების ანალიზისთვის, computer vision-ში ობიექტების ამოცნობისთვის და ა.შ.

Reinforcement learning

ეს მეთოდი ხშირად გამოიყენება რობოტოტექნიკაში,თამაშებში და ნავიგაციაში. Reinforcement learning(RL) -ში ალგორითმი გამოცდილებისა და შეცდომების ხარჯზე ადგენს იმ ქმედებას რომელსაც ყველაზე კარგი შედეგი ქონდა. ამ ტიპის აგორითმები სამი ძირითადი ტიპისგან შედგებიან: აგენტი(რაც სწავლობს ან გადაწყვეტილებებს იღებს), გარემო(ყველაფერი რასთანაც აგენტს კავშირი აქვს) და ქმედებები(რისი გაკეთებაც შეუძლია აგენტს). მეთოდის მიზანია დროის მოცემულ მონაკვეთში აგენტმა აირჩიოს ისეთი ქმედებათა თანმიმდევრობა, რომ მაქსიმალურად დადებითი შედეგი მიიღოს. აგენტი გაცილებით მალე მიაღწევს სასურველ შედეგს თუ სწორ ტაქტიკას აირჩევს ასე რომ RL-ის მიზანია ისწავლოს საუკეთესო ტაქტიკა

მანქანური სწავლების მაგალითები

1. მას იყენებენ ძებნის ალგორითმებში ისეთი კომპანიები როგორებიც არიან google, yahoo, microsoft(bing) და სხვა.
2. რეკლამების შერჩევისას ვებ გვერდებსა და მობილურ მოწყობილებებში. მაგალითად google AdSense
3. Text-based sentiment analysis ადგენს მოსაუბრის და მწერლის განწყობას ამათუიმ საკითხთან დაკავშირებით.
4. Credit scoring აბრუნებს ქულას რომელის მიხედვითაც ბანკმა ან სხვა მსგავს ორგანიზაციამ უნდა გადაწყვიტოს გაცეს თუ არა სესხი კონკრეტულ პიროვნებასა თუ ორგანიზაციაზე.
5. Network intrusion detection. სისტემაში არასანქციონალური შესვლის მცდელობას აფიქსირებს.
6. Pattern and image recognition. აქვს ფართო გამოყენება:
   6.1. მაგალითად მანქანის ნომრების ამოცნობა ციფრული გამოსახულებიდან (ANPR)
   6.2. სახის ამოცნობა (facial recognition)
   6.3. Pattern recognition-ის ნაწილში classification of text into several categories და speech recognition
   6.4. ხელნაწერის ციფრულ ფორმატში გადაყვანა და სხვა
7. წერენ ბოტებს რომლებიც დამოუკიდებლად პასუხობენ კლიენტს კითხვებზე.

ჩამოთვლილის გარდა კიდევ უამრავი გამოყენება არსებობს მეცნიერების ამ დარგისთვის.

დამხმარე ბიბლიოთეკები და არსებული სერვისები

დღესდღეისობით უკვე შექმნილია მრავალი ბიბლიოთეკა მეცნიერების ამ დარგში გამოსაყენებლად. ბიბლიოთეკებში იმპლემენტირებული არის ძირითადი ალგორითმები.

ოპტიმიზირებულია ამ ალგორითმებში გამოყენებული გამოთვლები. ეს კი საშუალებას გვაძლევს მეტი ვიფიქროთ უშუალოდ პრობლემაზე და მოდელის შედგენაზე.
ეს ბიბლიოთეკები დაწერილია სხვადასხვა ენებისთვის როგორიცაა python, R, C# matlab და ა.შ
ჩამოვთვალოთ რამდენიმე ბიბლიოთეკა:

1. scikit learn
2. shogun
3. mlib
4. ConvNetsJs

ასევე არსებობს სერვისები რომლებშიც უკვე იმპლემენტირებულია კონკრეტული პრობლემები. მათი გამოყენებით შეიძლება მანქანური სწავლების ცოდნის გარეშე მივიღოთ ის შედეგი რაც გვინდა.

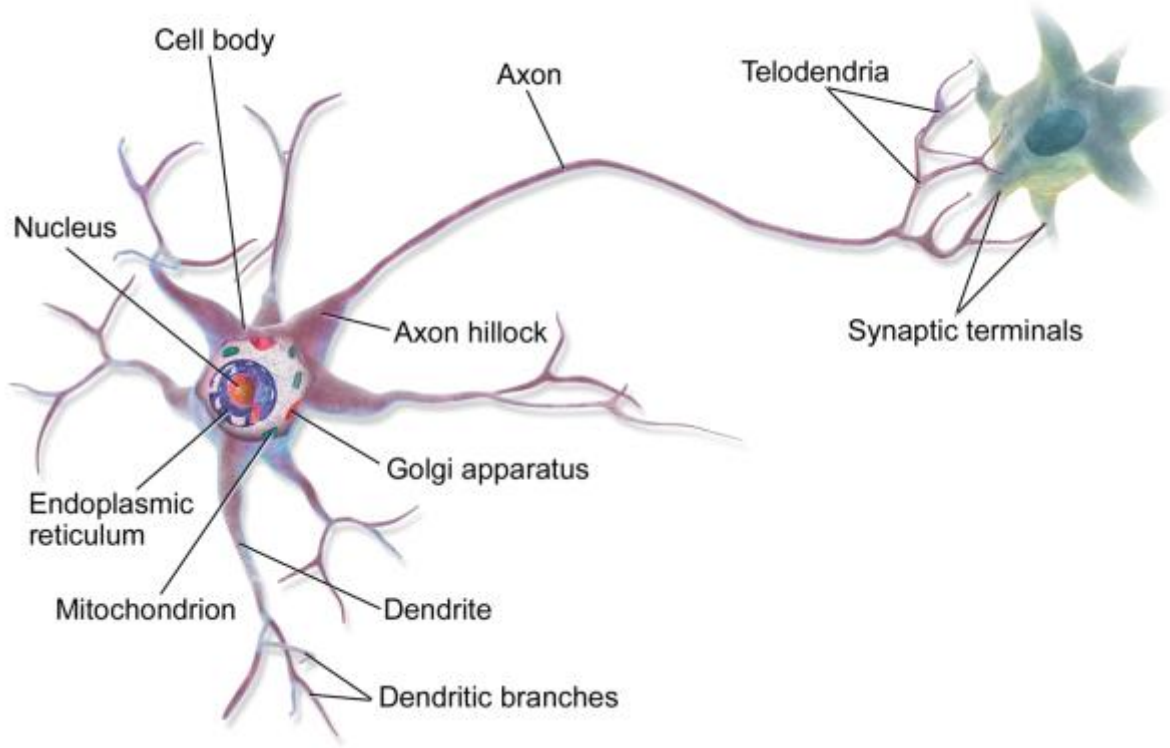
ასეთი სერვისებია:

1. Google prediction api
2. Amazon machin learning
3. Microsoft cognitive services

ნეირონული ქსელები

ნეირონული ქსელის წარმოდგენა

ნეირონული ქსელი (Artificial Neural Network) წარმოადგენს მანქანური სწავლების ალგორითმების ერთ-ერთ განაყოფს, რომელიც მიმართულია ადამიანის ტვინის მსგავსი ხელოვნური ქსელის სიმულირების მოდელირებისკენ. ტვინი შედგება უამრავი (10^{11}) სპეციფიკური უჯრედისგან - ნეირონისგან. ისინი შედგებიან თავად უჯრედისგან (სომა), და მისი წანაზარდებისგან - დენდრიტი და აქსონი. ეს 2 უკანასკნელი წარმოადგენს სინაპტიკური მუხტების მიმღებ/გადამცემთ - დენრდიტი მიიღებს, ხოლო გრძელი აქსონი ნეირონის უჯრედიდან გამოიტანს და გადასცემს სხვა ნეირონს, მისი დენდრიტის გავლით (ან პირდაპირ სომაზეც). ანუ ტიპიური სინაფსი გამოიყურება შემდეგნაირად:



სურ. 1

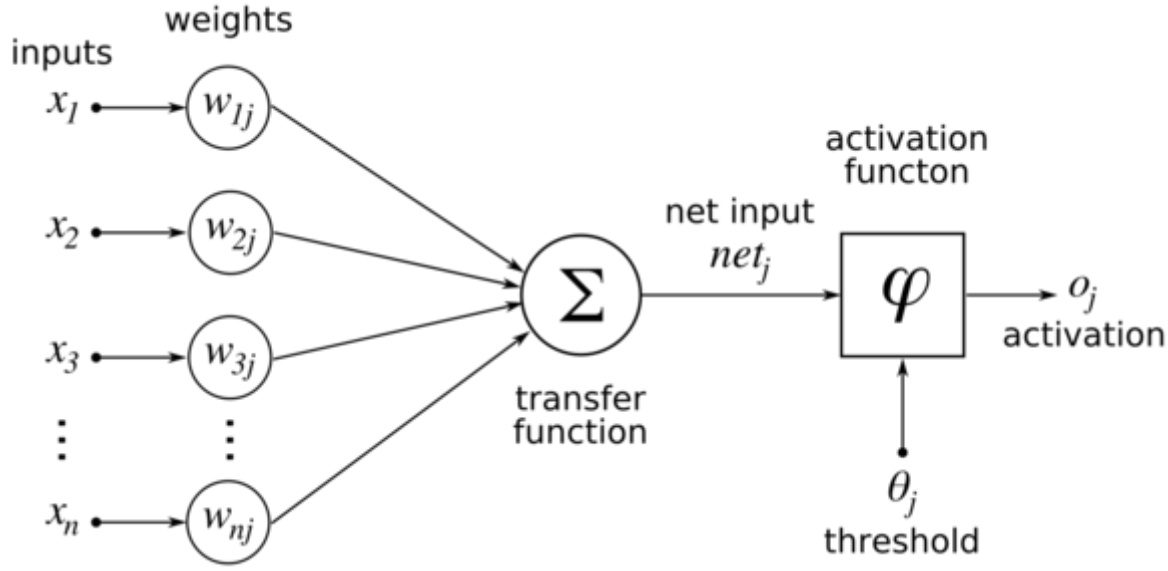
ტვინი შეიძლება მივიჩნიოთ ასეთი ნეირონების მრავალშრიან ციკლურ სტრუქტურად, სადაც გარე სამყაროს გამღიზიანებლებიდან (შემავალი ფენა) მიღებული იმპულსები გადაეცემა ტვინის ქერქს (შუალედურ ფენას), სადაც ხდება მათი დამუშავება და შემდეგ ისევ გარე სამყაროს დაუბრუნდება სამოქმედო შედეგი.

მათემატიკური მოდელი

ხელოვნური ნეირონული ქსელი ახდენს ტვინის მოდელირებას, რომელიც აერთიანებს ხელოვნურ ნეირონებს. ასეთ მოდელს შეუძლია ინფორმაციის მიღება გარედან, მისი დამუშავება და შედეგის გამოტანა.

როგორც პირველი სურათიდან ჩანს, ნეირონს შეიძლება გააჩნდეს მრავალი ინფორმაციის მიმღები (დენდრიტი) და ერთი გამომავალი წერტილი (აქსონი). ასევე, ხელოვნურ ნეირონში

მათემატიკური კუთხით, ნეირონულ ქსელი ასრულებს რამდენიმე $x(1), x(2) \dots x(n)$ შემავალი სიგნალების არაწრფივ გარდაქმნას y გამომავალ სიგნალად. ინფორმაციის დამმუშავებელი ნეირონის სტრუტურა წარმოდგენილია შემდეგნაირად.



სურ. 2

ის შედგება:

- შემავალი ინფორმაცია - $x(1), x(2) \dots x(n)$
- სინაფსური წონები - $w(1j), w(2j) \dots w(nj)$. თავიდან ისინი შემთხვევითად აიღება, ხოლო შემდეგ უკუგანვრცობის(Backpropagation) ალგორითმის გამოყენებით ხდება მათი კორექტირება.
- გადაცემის ფუნქცია - კრებს შემავალი ინფორმაციის წონებს.

- აქტივაციის ფუნქცია - ფუნქცია რომელიც ითვლის ნეირონის გამომავალ ინფორმაციას, როგორც წესი იღებენ არაწრფივ ფუნქციას [-1,1] ანდ [0,1] შუალედში. მოცემულ ნაშრომში აქტივაციის ფუნქციად აღებულია ჰიპერბოლური ტანგენსი.

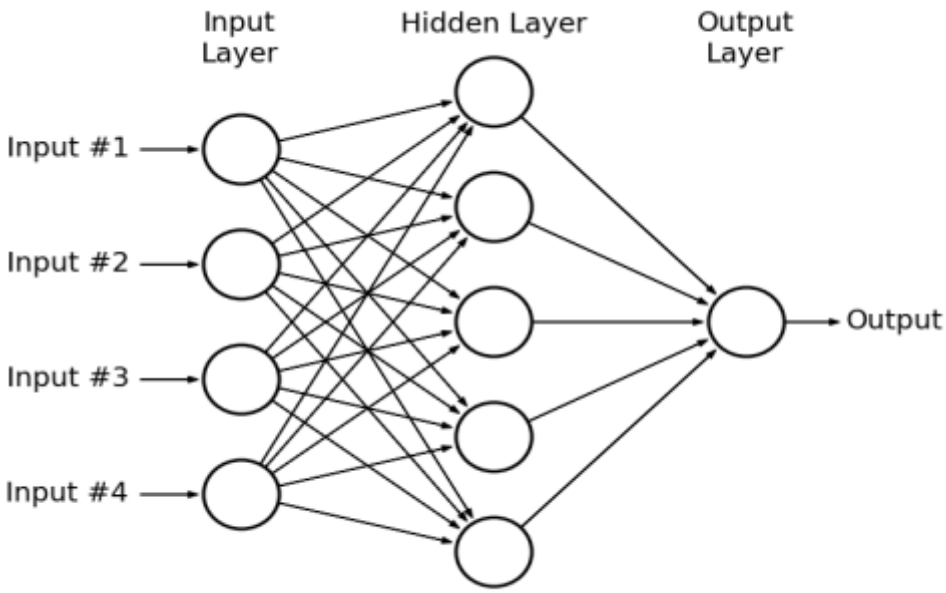
მათემატიკურად ნეირონი გამოისახება შემდეგნაირად:

$$u_k = \sum_{i=0}^{n} w_{ik} x_i$$

$$y_k = f(u_k + b_k)$$

სადაც b_k არის ეგრედ წოდებული ბიასი, y_k გამომავალი სიგნალი და f აქტივაციის ფუნქცია.

აღნიშნული მოდელი აღწერს 1 ნეირონს, მაგრამ ქსელში რამდენიმე ნეირონისგან შემდგარი ფენა შეიძლება შეიქმნას, სადაც წინა ფენის გამომავალი სიგნალები შეიქმნებიან შემდეგი ფენის შემავალ მონაცემებად.



სურ. 3

ნეირონული ქსელი შეიძლება შედგებოდეს 3 ძირითადი ნაწილისგან.

- Input layer ფენა რომლიდანაც ხდება საწყისი მონაცემების შეტანა. ნეირონებს ამ ფენაში ეწოდება input neurons.
- Output layer ფენა რომელიც გვიბრუნებს შედეგს(ამ შემთხვევაში ეს ფენა შედგება 1 ნეირონისგამ თუმცაღა ეს აუცილებლობას არ წარმოადგენს სხვა ქსელში output layer-ი შეიძლება შედგებოდეს 1-ზე მეტი ნეირონისგანაც). ნეირონებს ამ ფენაში ეწოდება output neurons
- შუა ფენა არის hidden layer. ტერმინ hidden layer-ს უკან დიდი მნიშვნელობა არ იმალება, ის უბრალოდ აღნიშნავს რომ ფენა არც input layer-ია და არ ც output layer-ი

უკუგანვრცობის ალგორითმი (Backpropagation)

ნეირონული ქსელი საინტერესო და განსხვავებული იმითაა, რომ მას შეუძლია დასწავლა, ისევე როგორც ნამდვილ, ცოცხალ ტვინს. დასწავლა მიმდინარეობს უკუგანვრცობის ალგორითმით.

თავდაპირველად ხდება პირდაპირი განვრცობა (Forwardpropagation), რის შედეგადაც გამოითვლება ქსელის გამომავალი მნიშვნელობა, შემდეგ ხდება ამ მნიშვნელობის რეალურთან შედარება და შეცდომების დათვლა:

$$\sum \frac{1}{2}(y_i - h_w(x_i))^2$$

სადაც y_i არის x_i შემავალი ინფორმაციისთვის რეალური გამომავალი მნიშვნელობა, ხოლო $h_w(x_i)$ არის პირდაპირი განვრცობის შედეგად მიღებული მნიშვნელობა.

იმისათვის, რომ წარმატებით განხორციელდეს დასწავლა, საჭიროა შეცდომები იყოს მინიმალური, ანუ უნდა მოვახდინოთ

$$J(w) = \sum \frac{1}{2}(y_i - h_w(x_i))^2$$

ფუნქციის მინიმიზაცია. გრადიენტული დაშვების ალგორითმით ყოველი w-თვის გამოვითვლით გრადიენტებს

$$\frac{\partial W}{\partial w_i} = -\sum_i (y_i - h_w(x_i)) x_{ji}$$

და განვაახლებთ მათ მნიშვნელობას

$$w_i = w_i - \frac{\partial W}{\partial w_i}$$

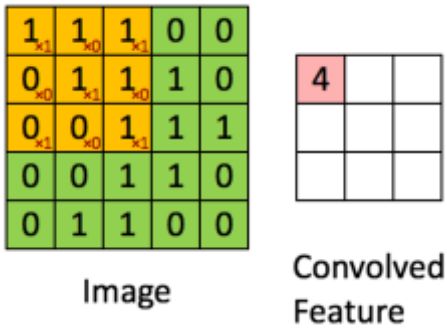
როდესაც ნეირონული ქსელი დაასრულებს დასწავლას, წონებს ექნებათ ისეთი მნიშვნელობები, რომ $J(w) = \sum \frac{1}{2}(y_i - h_w(x_i))^2$ იქნება 0-თან მიახლოვებული.

კონვოლუციური ნეირონული ქსელები

არქიტექტურა

ჩვეულებრივი ნეირონული ქსელებისგან განსხვავებით, კონვოლუციური ნეირონული ქსელების დაფარული დონეები გვხვდება რამდენიმე სახის:

- კონვოლუციური დონე - წარმოადგენს ნეირონების სიმრავლეს, რომლის განზომილებაც გაცილებით ნაკლებია წინა დონის განზომილებაზე. კონვოლუციური დონის შემავალი ინფორმაცია მიიღება წინა დონეზე კონვოლუციით, ანუ აიღება კონვოლუციის განზომილების ფილტრი, და ხდება მისი მოძრაობა წინა დონეზე მანამ, სანამ ის სრულიად არ

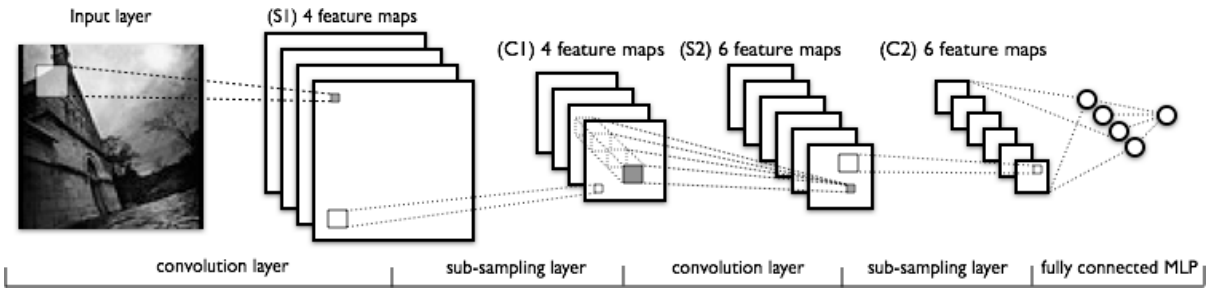


დაიფარება. სურ. 4

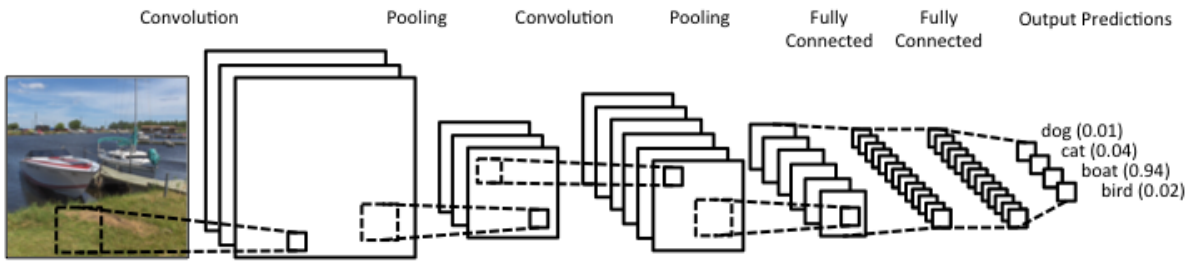
სურათში ჩანს 5x5 განზომილების შემავალი დონე და 3x3 განზომილების კონვოლუციური ფილტრი, კონვოლუციური ფილტრის თითოეული ელემენტი მიიღება შემავალ დონეზე 3x3 განზომილების კონვოლუციით.

- Max-Pooling დონე - თითოეული კონვოლუციური დონის შემდეგ, შესაძლოა გვქონდეს Max-pooling დონე. ამ დონეში ხდება კონვოლუციური დონის ელემენტებიდან მაქსიმუმის ამორჩევა, შესაბამისად ვიღებთ თითო კონვოლუციურ ფილტრზე ერთ ელემენტს.
  Max pooling დონე საშუალებას გვაძლევს შევამციროთ ქსელის განზომილება, რაც ამარტივებს გამოთვლებს. ამ პროცესს სხვანაირად subsampling-საც უწოდებენ.

- სრულად შეერთებული(Fully connected) დონე - საბოლოოდ, რამდენიმე კონვოლუციური და max-pooling დონის შემდეგ მოდის სრულად შეერთებულ დონე, რომელიც წარმოადგენს ჩვეულებრივ ნეირონულ ქსელებში დაფარული დონის ანალოგს.



სურ. 5



სურ. 6

მათემატიკური მოდელი

ისევე, როგორც ჩვეულებრივ ნეირონულ ქსელებში, კონვოლუციურ ნეირონულ ქსელებშიც გვაქვს პირდაპირი და უკუგანვრცობა(Forward propagation and Backward propagation).

პირდაპირი განვრცობა, იმის მიხედვით თუ რომელ დონეზე ვიმყოფებით განსხვავებულად მიმდინარეობს.

პირდაპირი განვრცობა კონვოლუციურ დონეში

დავუშვათ შემავალი დონის განზომილებაა $N \times N$, რომელსაც მოჰყვებაგანზომილების კონვოლუციური დონე. თუ კონვოლუციურ ფილტრების განზომილება იქნება $m \times m$, შედეგად მივიღებთ $N - m + 1 \times N - m + 1$ განზომილებისკონვოლუციურ დონეს. ∂ კონვოლუციურ დონეში X_{ij} ნეირონი მიიღება შემდეგი ფორმულით:

$$X_{ij}^{\partial} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} w_{ab} y_{(i+a)(j+b)}^{\partial-1}$$

პირდაპირი განვრცობა max-pooling დონეში

max-pooling-ში დასწავლა არ მიმდინარეობს, როგორც ზემოთ აღვნიშნეთ, მისი გამოყენების მიზანია განზომილების შემცირება ქსელში. თუ კონვოლუციური დონის განზომილებაა $m \times m$, ხოლო შემავალი დონის განზომილება - $N \times N$ შედეგად მივიღებთ $\frac{N}{m} \times \frac{N}{m}$ დონეს, სადაც თითოეული $m \times m$ ბოლკში აირჩევა მაქსიმალური მნიშვნელობა.

პირდაპირი განვრცობა სრულად შეერთებულ დონეში

სრულად შეერთებულ დონეში პირდაპირი განვრცობის ალგორითმი ანალოგიურია ჩვეულებრივ ნეირონულ ქსელში პირდაპირი განვრცობის ალგორითმის.

$$u_k = \sum_{i=0}^{n} w_{ik} x_i$$

$$y_k = f(u_k + b_k)$$

სადაც b_k არის ეგრედ წოდებული ბიასი, y_k გამომავალი სიგნალი და f აქტივაციის ფუნქცია.

უკუგანვრცობა კონვოლუციურ ნეირონულ ქსელში

იმისათვის რომ ქსელმა მოახერხოს დასწავლა, საჭიროა პირდაპირი განვრცობის შემდეგ წონების კორექტირება. ამ შემთხვევაშიც ვახდენთ

$$J(w) = \sum \frac{1}{2}(y_i - h_w(x_i))^2$$

ფუნქციის მინიმიზაციას. შემდეგ კი გრადიენტული დაშვების ალგორითმით ყოველი w-სთვის გამოვითვლით გრადიენტებს

$$\frac{\partial W}{\partial w_i} = -\sum_i (y_i - h_w(x_i)) x_{ji}$$

და განვაახლებთ წონების მნიშვნელობას

$$w_i = w_i - \frac{\partial W}{\partial w_i}$$

შედეგად მივიღებთ შეცდომების 0-თან მიახლოებულ მნიშვნელობას.

კონვოლუციური ნეირონული ქსელების გამოყენება ტექსტების კლასიფიკაციისთვის

კონვოლუციური ნეირონული ქსელების ტექსტების კლასიფიკაციისთვის გამოსაყენებლად, საჭიროა დავადგინოთ შემავალი დონის ზომა, ამისათვის ვპოულობთ ტექსტებში არსებულ ყველაზე გრძელ წინადადებას და შემდეგ ყველა დანარჩენ წინადადებას იმდენჯერ ვუმატებთ სპეციალურად არჩეულ სიტყვას, რომ მათი სიგრძე იყოს ერთი და იგივე. ამასთან ეს სპეციალურად არჩეული სიტყვა თავდაპირველად არ უნდა შედიოდეს არცერთ ტექსტში. ამ ოპერაციის შემდეგ ტექსტებში ყველა წინადადება იქნება ერთი და იგივე სიგრძის, რომელიც წარმოადგენს შემავალი დონის ზომას.

```
def pad_sentences(sentences, padding_word="<PAD/>"):
    """
    Pads all sentences to the same length. The length is defined     by the longest sentence.
    Returns padded sentences.
    """
    sequence_length = max(len(x)  for x in sentences)
    padded_sentences = []
    for i in range(len(sentences)):
        sentence = sentences[i]
        num_padding  = sequence_length  - len(sentence)
        new_sentence = sentence + [padding_word] * num_padding
        padded_sentences.append(new_sentence)
    return padded_sentences
```

შემდეგ ეტაპზე ტექსტებიდან ვარჩევთ სიტყვების სიმრავლეს და ამ სიმრავლის თითოეულ ელემენტს შევუსაბამებთ რიცხვს, რაც შესაძლებლობას გვაძლევს თითოეული წინადადება წარმოვადგინოთ ვექტორის სახით. მიიღება ეგრედ წოდებული ლექსიკონი.

```
def build_vocab(sentences):
    """
    Builds a vocabulary mapping from word to index based on the sentences.
    Returns vocabulary mapping and inverse vocabulary mapping.
```

```
    """
    # Build vocabulary
    word_counts = Counter(itertools.chain(*sentences))
    # Mapping from index to word
    vocabulary_inv = [x[0] for x in word_counts.most_common()]
    vocabulary_inv = list(sorted(vocabulary_inv))
    # Mapping from word to index
    vocabulary = {x: i for i, x in enumerate(vocabulary_inv)}
    return [vocabulary, vocabulary_inv]


def build_input_data(sentences, labels, vocabulary):
    """
    Maps sentencs and labels to vectors based on a vocabulary.
    """
    x = np.array([[vocabulary[word] for word in sentence] for sentence in sentences])
    y = np.array(labels)
    return [x, y]
```

შემდეგი ბიჯი არის კონვოლუციების რაოდენობებისა და მათი ზომების არჩევა. ამ ნაშრომში ავარჩიეთ სამი კონვოლუცია რომლელთა ზომებია 3, 4 და 5. რაც უფრო დიდია კონვოლუციების ზომა და რაოდენობა, მით უფრო დიდი დრო სჭირდება მოდელის გენერაციას, სამაგიეროდ დიდია მისი სიზუსტე. როდესაც შემავალ დონეზე მივიღებთ ვექტორს, საშუალება გვექნება 3, 4 და 5 სიტყვის ჩასმა კონვოლუციურ ფილტრში, სწორედ ეს გვაძლევს კონტექსტის შენარჩუნების საშუალებას.

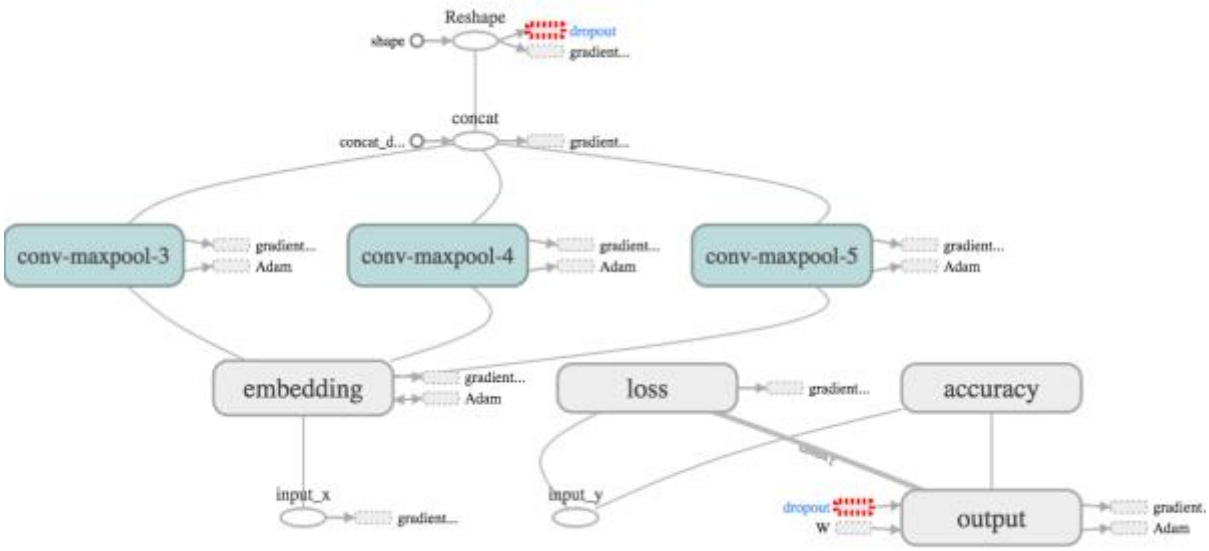
როდესაც შემავალი ინფორმაცია გარდაქმნილია ვექტორებად და არჩეულია კონვოლუციები, ვატარებთ პირდაპირ და უკუგანვრცობას. მოდელის მიღების შემდეგ შეგვიძლია კლასიფიკაცია გავუკეთოთ ნებისმიერ ტექსტს. თუ ამ ტექსტში არსებული რომელიმე სიტყვა არაა მეორე ბიჯზე მიღებულ ლექსიკონში, ამ სიტყვას ჩავანაცვლებთ პირველ ბიჯზე არჩეული სპეციალური სიტყვით.

```
array = []
for word in new_sentence:
    try:
        word_vector=vocabulary[word]
    except KeyError:
```

```
            word_vector=vocabulary["<PAD/>"]
        array.append(word_vector)
    x=np.array([array])
```

აღნიშნული ნეირონული ქსელის ასაგებად, გამოვიყენეთ პროგრამირების ენა Python-ზე აგებული ფრეიმვორკი: Tensorflow (https://www.tensorflow.org).
ეს ფრეიმვორკი საშუალებას გვაძლევს გრაფიკულად დავინახოთ ქსელი და მოდელის გენერაციისას თვალი მივადევნოთ სიზუსტისა და შეცდომების ცვლილებებს:

მიღებული ნეირონული ქსელი:



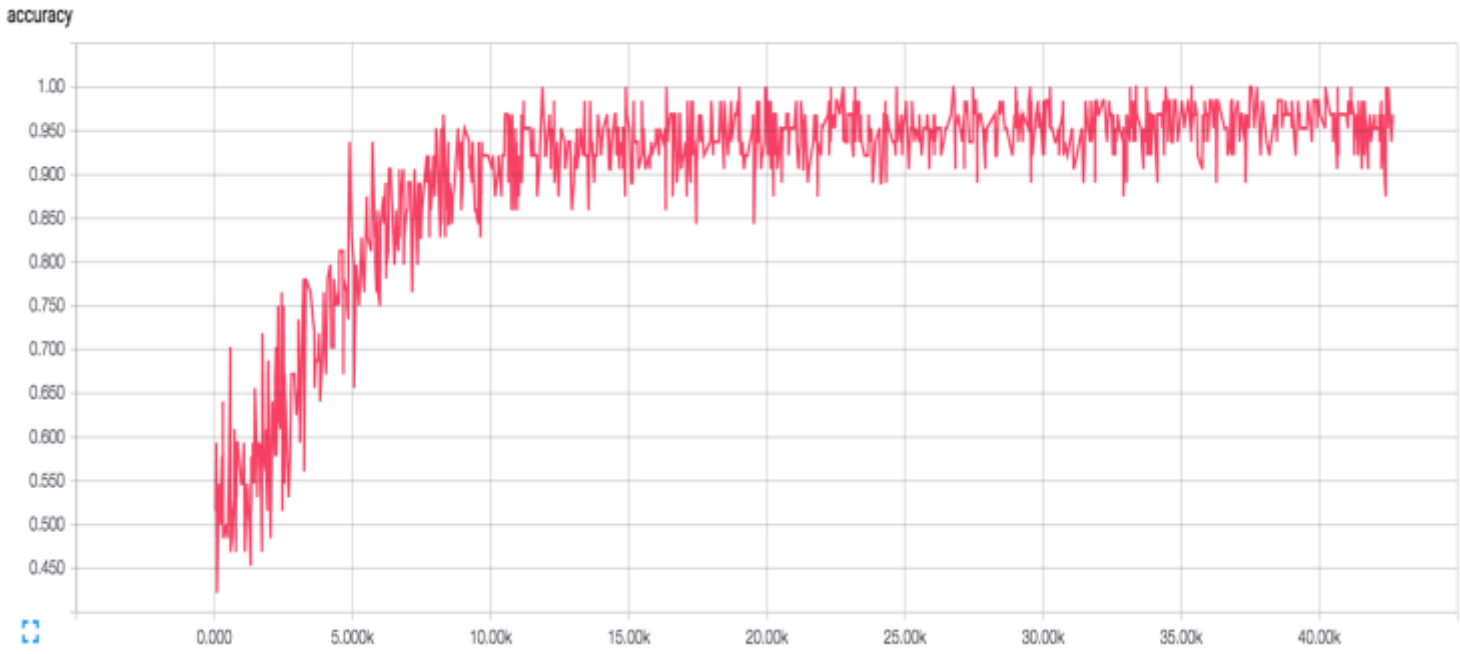
სურ. 7

- embedding - შემავალ დონეზე მიღებული წინადადების ვექტორებად გარდაქმნა ლექსიკონიდან.
- conv-maxpool-3, conv-maxpool-4, conv-maxpool-5, - კონვოლუციური დონეები შესაბამისი maxpooling დონეებით.
- Loss - უზუსტობა

- Accuracy - სიზუსტე
- Output - გამომავალი დონე

სურ. 8

X ღერძზე მოცემულია წინადადებების რაოდენობა, ხოლო Y-ზე - ცდომილება. თავდაპირველად გვაქვს ძალიან დიდი ცდომილება, შემდეგ კი უკუგანვრცობის შედეგად კორექტირდება წონები და უზუსტობაც იკლებს. როგორც გრაფიკიდან ჩანს, 40000 წინადადების შემდეგ უზუსტობა თითქმის ნულია.



სურ. 9

X ღერძზე მოცემულია წინადადებების რაოდენობა, ხოლო Y-ზე - სიზუსტე. თავდაპირველად სიზუსტე თითქმის ნულია, შემდეგ კი უკუგანვრცობის შედეგად კორექტირდება წონები და 40000 წინადადების შემდეგ ვიღებთ თითქმის 100%-იან შედეგს.

დასკვნა

შედეგად გვაქვს ნეირონული ქსელის სტრუქტურა, რომელიც შეგვიძლია მოვარგოთ ნებისმიერ ტექსტურ ინფორმაციას, ავაგოთ მოდელი და ჩავატაროთ სენტიმენტ ანალიზი. გვაქვს აღწერილი ამ პროცესის მათემატიკური ნაწილი, რაც საშუალებას გვაძლევს უფრო სიღრმისეულად გავიგოთ და გავაანალიზოთ მიმდინარე პროცესები. აღწერილია ალგორითმის მოქმედებების პრინციპი, მიდგომები, დადებითი და უარყოფითი მხარეები. ამის საფუძველზე ჩვენ შევძლებთ გავაანალიზოთ და ჩამოვაყალიბოთ სხვა მსგავსი ტიპის პრობლემები, საჭიროებისამებრ ცვლილებები შევიტანოთ ალგორითმსა და ალგორითმში გამოყენებულ მათემატიკურ ნაწილში. მოვარგოთ ნაშრომის ძირითად ტექსტში არსებული პროცესები ახალ პრობლემებს.

მაგალითისთვის ნაშრომში აღწერილია კლსაიფიკაციის პრობლემის მოგვარების ერთ-ერთი მეთოდი. კლასიფიკაციის პრობლემათა ჯგუფი ერთ-ერთი ყველაზე დიდი ჯგუფია მანქანური დასწავლის პრობლემებს შორის. იმავე ალგორითმის მოდიფიკაციით შეგვიძლია გავაკეთოთ სისტემა, რომელიც ტექსტის კლასიფიკაციას მოახდენს არა მხოლოდ ორი, არამედ მრავალ კლასში. ამისათვის ჩვენს ნეირონულ ქსელს პარამეტრში `num_classes` უნდა გადავცეთ არგუმენტი 3 და ახალი მოდელის შესაქმნელად უნდა მოვიძიოთ 3 კლასში დანაწილებული ინფორმაცია.

შემდგომში სასურველია სისტემა გადაკეთდეს სერვისად, რაც საშუალებას მისცემს სხვა სისტემებს მოახდინონ აღნიშნული სისტემის ინტეგრაცია და გამოიყენონ მისი ფუნქციონალი სხვადასხვა ტექსტური კლასიფიკაციის პრობლემების გადაჭრაში. ამისათვის საჭიროა შეიქმნას ვებ სერვისი, რომელიც გამოიყენებს ნაშრომში მიღებულ მოდელს, სერვისს შემავალ პარამეტრად გადაეცემა წინადადება და დააბრუნებს ამ წინადადების კლასს.

გამოყენებული ლიტერატურა

- Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), 1746–1751.
- Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. Acl, 655–665.
- Santos, C. N. dos, & Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In COLING-2014 (pp. 69–78).
- Johnson, R., & Zhang, T. (2015). Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. To Appear: NAACL-2015, (2011).
- Johnson, R., & Zhang, T. (2015). Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding.
- Wang, P., Xu, J., Xu, B., Liu, C., Zhang, H., Wang, F., & Hao, H. (2015). Semantic Clustering and Convolutional Neural Network for Short Text Categorization. Proceedings ACL 2015, 352–357.
- Zhang, Y., & Wallace, B. (2015). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification,
- Nguyen, T. H., & Grishman, R. (2015). Relation Extraction: Perspective from Convolutional Neural Networks. Workshop on Vector Modeling for NLP, 39–48.
- Zhang, X., & LeCun, Y. (2015). Text Understanding fromScratch. arXiv E-Prints, 3, 011102.