Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Intelligens akusztikus objektumazonosítás otthoni környezetben

Készítette
Ossik László

Konzulens
Dr. Orosz György

2018

TARTALOMJEGYZÉK

HALLGATÓI NYILATKOZAT

Alulírott Ossik László, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2018. 12. 11.

.....

Ossik László

Kivonat

Manapság egyre nagyobb teret hódítanak azon alkalmazások, amelyek a környezetükben észlelt hangok, változások alapján képesek döntéseket hozni, feladatokat ellátni. Ilyen például az Amazon által fejlesztett Alexa is, aki az internetes vásárlások során felmerülő kérdésekben segíti ki az ügyfeleit. Az IoT (Internet of Things) területén különböző szenzorokat használnak arra a célra, hogy a kiszemelt környezetet vizsgálni tudják. A detektált események hatására pedig valamilyen végső feladat kerül elvégzésre.

A szakdolgozat keretén belül egy olyan rendszer megvalósítása volt a cél, amely képes az otthonunkban előforduló hangforrásokat elkülöníteni valamilyen osztályozó algoritmus felhasználásával.

A dolgozatban bemutatásra kerül a kialakított adatbázis felépítése, a Matlabban megvalósított alkalmazás rendszerterve, hat algoritmus, amely tulajdonságvektorok előállítására szolgál (MFCC, LPC, FFT, Cepstrum, Reflexiós együtthatók és Mel-spektrum). Betekintést nyerhetünk a neurális hálózatok világába. Bemutatom felépítésüket, működésüket. Majd végezetül a működő rendszer eredményeit vizsgáljuk meg.

Abstract

Nowadays, there are more and more applications, which are able to make decisions, and carry out tasks based on the heard voices and changes from the environment. For example Alexa who was developed by Amazon. Her task is to help her clients through questions that arise on internet purchases. Different sensors are used on the Internet of Things to explore the observed environment. As a result of the perceived events, final tasks are performed.

The project's goal was to implement a system which is capable of isolating the sound sources in our home using some sorting algorithm.

The thesis presents the structure of the created database, the system plan of the application which was implemented in Matlab, six algorithms were used for producing different feature vectors (MFCC, LPC, FFT, Cepstrum, Reflection coefficients and Mel-spectrum). We can gain insight into the neural network, how do they work, how can we create our own, and how does the learning of the neural network really work. Finally, we will examine the results of the system.

1. Bevezetés

Az ember rengeteg érzékszervvel rendelkezik, amely segítségével a környezetében képes tájékozódni. Bizonyos helyzetekben, az események nem a szemünk előtt zajlódnak le, ekkor viszont csak más érzékszervünkre hagyatkozhatunk, mint ahogy járművezetés közben vizsgáljuk szemünkkel a körülöttünk lévő forgalmat, fülünkkel párhuzamosan figyeljük a motor járását, a közeledő szirénázó mentő hangját. Napjainkban az otthoni környezetben biztonsági célokra főként kamerákat alkalmaznak, amelyek képesek az eszköz által látottakat rögzíteni. De mi a helyzet azokkal az esetekkel, amikor nem a megfigyelő előtt zajlanak a történések? E probléma kiküszöbölésére alkalmazhatunk elszórtan mikrofonokat lakásunkban, amelyek folyamatosan figyelik annak minden rezzenését.

Ezek az alkalmazások elláthatnak időseket segítő otthoni felügyeleti vagy kényelmi funkciókat is. A szakdolgozat témáját is egy hasonló gondolatmenet hozta létre, ahol a otthonunkban történő eseményeket próbáljuk követni. Képzeljük el azt a helyzetet, hogy családos emberek vagyunk, a gyermekünk a házban tartózkodik, de nekünk el kell sietnünk egy közeli élelmiszerboltba pár percre. Ezalatt az idő alatt a rendszer folyamatosan tájékoztat bennünket, a házban zajló történésekről. Minden egyes észlelt esemény naplózásra kerül, ahonnan visszatudjuk követni, mi és mikor történt.

Feladatomban egy olyan rendszert hoztam létre, amely képes különböző hangosztályokból származó hangokat felismerni, egymástól azokat megkülönböztetni. A szoftver Matlabban került implementálásra, mivel a Matlab által nyújtott jelfeldolgozási lehetőségek vonzóbbak voltak számomra, mint a más környezeteknél megtapasztaltak.

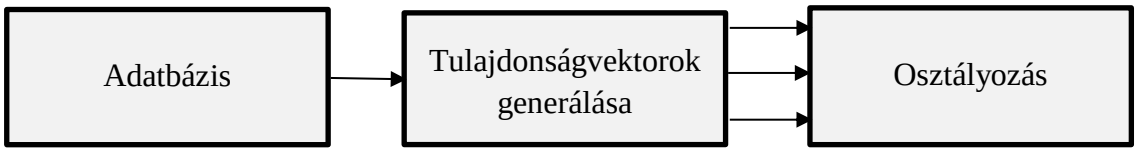
Egy valódi alkalmazás valós idejű adatgyűjtéssel és kiértékeléssel dolgozik. A szakdolgozat keretein belül nem egy valódi alkalmazás létrehozása volt a cél, hanem egy prototípus rendszeré, amely képes a hangfelvételek offline kiértékelésére. A továbbiakban bemutatásra kerül a kialakított rendszer, a tulajdonságvektorok generálására alkalmas algoritmusok és a végső eredmények.

2. Rendszerterv

A rendszer megtervezése egy logikailag átfogó kép megteremtésével kezdődött. Ebben a fázisban megmutatkoznak a rendszer szempontjából kritikusabb pontok. Melyek azok a részek, amik kisebb vagy nagyobb mértékben emésztik fel a fejlesztésre szánt időtartamot. Ennek segítségével az esetleges felmerülő problémákkal is hamarabb szembesülünk.

A rendszer segítségével egy osztályozási probléma megoldása a cél. Ehhez szükséges megfelelő mennyiségű és különböző osztályba csoportosítható hanganyag, algoritmusok, amelyek a hanganyag lényegesebb tulajdonságait emelik ki, illetve egy osztályozási eljárás, amely a rendszer válaszát szolgáltatja. Ideális esetben a rendelkezésre álló adatbázis nagyon sok mintát tartalmaz, így csupán a nyers mintákkal való tanítás is jó eredményekhez vezethet. A kis méretű adatbázisok esetében viszont szükséges némi támogatás az osztályozó algoritmusnak. Ezt valósítják meg a tulajdonságvektorok generálására szolgáló algoritmusok.

A főbb blokkok definiálása után, azokat funkcionalitásokkal, feladatokkal ruházzuk fel, figyelembe véve a követelmények teljesítését. A főbb blokkok a 2-1. ábrán láthatók.



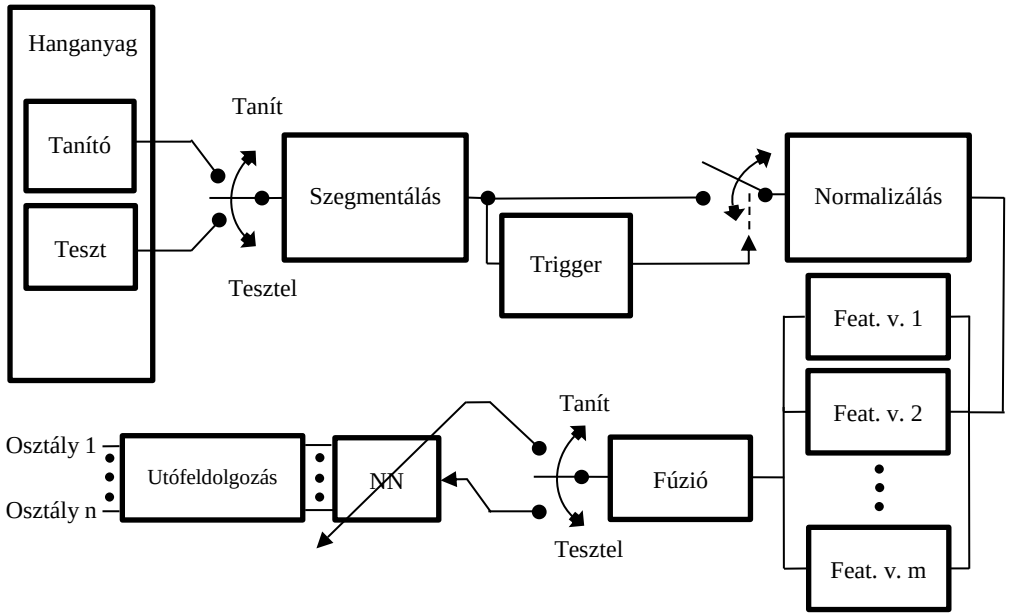
2-1. ábra – A rendszerterv főbb blokkjai

- **Adatbázis**: Megfelelő mennyiségű és minőségű hanganyagok tárolására szolgál. A tulajdonságvektorokat generáló algoritmusok innen kerülnek kiszolgálásra.
- **Tulajdonságvektorok generálása**: Az adatbázisból beolvasott adatokból a fontosabb tulajdonságok kiemelése (periodikusság, impulzus szerű változások, főbb frekvenciakomponensek). Az alábbi tulajdonságvektor generáló algoritmusok kerülnek felhasználásra:
  - FFT (Fast Fourier Transform)
  - Mel Spektrum
  - Cepstrum
  - MFCC (Mel Frequency Cepstral Coefficients)
  - Reflexiós együtthatók

- LPC (Linear Predictive Coding)

- **Osztályozás**: Feladata a tulajdonságvektorok által szolgáltatott információ megtanulása, illetve tetszőleges bemeneti mintához megtalálni a mintához tartozó osztályt.
  - KNN
  - Neurális hálózat

A 2-2. ábrán a főbb blokkok további részfeladatokra való bontása látható.



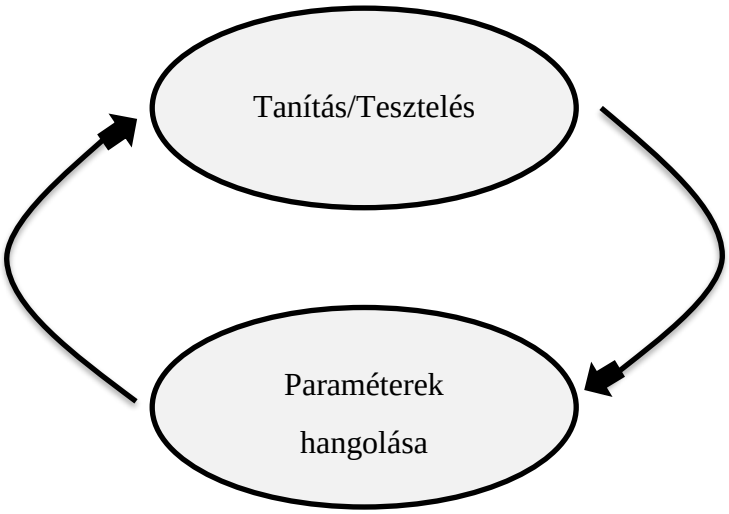
2-2. ábra – Kifejtett rendszerterv

Az adatbázis blokk áll egy tanító, illetve teszthalmazból. A teszthalmaz szintén tovább van bontva két teszthalmazra, amelynek oka majd később kerül kifejtésre. Két fázist különböztetünk meg a rendszer szempontjából:

- Tanulási fázis: Ezen a ponton képes a neurális hálózat a belső paramétereit megfelelően állítva, rögzíteni a tulajdonságvektorokban hordozott információkat.
- Tesztelési fázis: A hálózat működését vizsgáljuk.

Az algoritmus blokkos adatfeldolgozást végez, ennek oka, hogy a tulajdonságvektort generáló algoritmus is blokkos adathalmazt vár. Lehetőség van triggerszint beállítására, amely segítségével a hanganyagban lévő információt nem hordozó csendes szakaszok kiszűrését valósíthatjuk meg. A normalizálás fontossága abban rejlik, hogy az adatbázisban lévő hangok nem mind megegyező hangerősségűek. Adott események nem mindig

megegyező távolságokban zajlanak a mikrofontól, így adott hangesemény esetén különböző hangerősségek lehetnek. Elkerülve azt, hogy a hangforrást hangerősség alapján osztályozzuk, egységnyi teljesítményre hozzuk. A következő részben az egyes tulajdonságvektorok generálása zajlik. A blokkban szereplő elnevezés (Feat. v.) a Feature vector rövidítése, amelynek magyar megfelelője: tulajdonságvektor. A legenerált vektorokat egységesítem, hogy könnyebben kezelhetőek legyenek, majd a korábban kiválasztott fázisnak megfelelően tanítom a hálózatot vagy tesztelem. A neurális hálózat által szolgáltatott felismerési arány javítható valószínűségi következtetések levonása alapján, így a hálózat kimenetét további feldolgozásnak vetem alá. A kimeneten pedig az osztályokba való tartozás százalékos eredménye látható.



2-3. ábra – Paraméterek beállításának folyamata

A rendszer kezdeti fázisában a paraméterek beállítása szükséges a megfelelő működés elérése érdekében. Ennek folyamata a 2-3. ábrán látható. A paraméterek hangolása mindaddig tart, míg az általunk elfogadott eredményt nem hoz létre a rendszer.

3. A munkafolyamat bemutatása

Ahogy az a rendszertervben látható volt, a kiindulási pont az adatbázis, amely szinte az egyik legfontosabb pontja a teljes rendszernek. Így az én munkám is az adatbázis megteremtésével kezdődött. Következő lépésként a szoftverkörnyezet megválasztása, és az abban történő implementációk végrehajtása volt a feladat. Szoftverkörnyezet tekintetében a Matlab-ra esett a választás. Ennek oka, hogy jelfeldolgozási szempontból erősen támogatott. Rengeteg olyan toolbox-al rendelkezik, amelyek megkönnyítik a szoftveres megvalósítást. A rendszer létrejötte után paramétereinek hangolása és tesztelése volt a cél a megfelelő működés elérése érdekében.

3.1. Adatbázis

Nem megfelelő mennyiségű és minőségű adatokkal a rendszer nem képes a klasszifikációt megfelelően végrehajtani. Kevés rendelkezésre álló adattal a rendszer nehezen alkalmazható általánosított helyzetekben. Az adatbázis megalkotását megelőzte a lehetséges osztályok definiálása. A tesztelést figyelembe véve az emberi fül számára hasonló, illetve elkülönülő osztályokra esett a választás, ezek alább láthatók:

- Csecsemő sírás (Baby crying)
- Fürdés közben keltett mozgások hangja (Bath water movement)
- Autó indítás (Car starting)
- Macskanyávogás (Cat meow)
- Kutyaugatás (Dog barking)
- Férfi/Női beszéd (Men/Women speaking)
- Borotválkozás (Razor sound)
- Tusolás (Showering)
- Porszívózás (Vacuum cleaner sound)

Az adatbázis olyan előre definiált osztályok hangmintáit tartalmazza, amelyek egy családi környezetben otthon előfordulhatnak. Annak ellenére, hogy manapság elég elterjedt ez a tématerület, rendelkezésre álló teljes adatbázis nagyon nehezen található. A férfi/női beszéd osztály mintáit a tanszékről kölcsönzött hangkártyával, és mikrofonnal sikerült rögzíteni. A felhasznált hangkártya a 3-1. ábrán látható. A rögzítéshez Audiacityt [1] használtam, amelyet felvételkor laptopon futtattam. A tartalma 50-50 férfi, illetve női hang. A többi osztályt az interneten rendelkezésre álló erőforrások [2] felhasználásával hoztam létre. Minden egyes elem .wav kiterjesztésű, így elkerültem az .mp3 fájloknál tömörítésekből fakadó adatvesztést.

3-1. ábra - M-Audio FastTrack Pro hangkártya

A későbbiekben látni fogjuk, hogy nagyon fontos szerepe van az adatbázis nagyságának. Minél több hanganyag áll rendelkezésre, annál több tanító minta készíthető, több a viszonyítási alap, amely segítségével a jó döntéseket az algoritmus meghozhatja.

A 3-1. táblázat tartalmazza, hogy a teljes adatbázis milyen hosszú hanganyagokból áll.

Csecsemő sírás	147 másodperc
Fürdés közben keltett mozgások hangja	144 másodperc
Autó indítás	366 másodperc
Macska nyávogása	152 másodperc
Kutya ugatása	251 másodperc
Férfi/Női beszéd	1540 másodperc
Borotválkozás	173 másodperc
Tusolás	238 másodperc
Porszívózás	159 másodperc

3-1. táblázat - Adatbázis

A későbbi tesztelést figyelembe véve, az adatbázist 3 részre osztottam. Az első 1/3-a tanítási, a második 1/3-a tesztelési, míg a harmadik 1/3-a további tesztelési célokra lett felhasználva. A két elkülönülő teszthalmaz szükségességének oka, hogy a neurális háló paramétereinek hangolása a teszt adatokra adott válasz alapján történik, és ezen felül szükséges még egy olyan teszthalmaz, amelyet nem használtam fel a hálózat paramétereinek változtatására. Ezzel valósabb képet kapva, hogy mennyire általánosítva dolgozik a neurális hálózat.

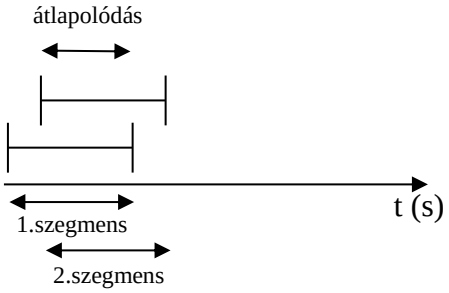
3.2. Szoftveres implementálás

A program indulása a megfelelő könyvtár megtalálásával indul, ahol rendelkezésre áll az adatbázis. Mivel a tulajdonságvektorok generálása egy hosszadalmas folyamat (a jelenlegi adatbázissal is körülbelül fél óra), ezért egy ellenőrzést hajtok végre, amely megvizsgálja, hogy korábban került-e már legenerálásra tulajdonságvektor. Ha igen és frissebb, mint az adatbázis utolsó módosításának ideje, akkor felhasználhatóak. Ha viszont valamely algoritmus által generált vektorok hiányoznak, vagy az adatbázist módosították, és azóta nem hoztak létre új vektorokat, akkor a tulajdonságvektorok legenerálása a következő lépés. Ezen a ponton állíthatók be a következő paraméterek:

- Szegmens hossz
- Átlapolódás nagysága
- Triggerszint

3.2.1. Szegmentálás

A tulajdonságvektorok generálását egy úgynevezett előfeldolgozás előzi meg. Ezen a ponton az adatbázisban szereplő hanganyagokat szegmensekre bontom. Ennek azért van nagy fontossága, mert így létrehozhatok a szegmensek között valamekkora átlapolódást. Aprólékosabban szemlélve a teljes hanganyagot több felhasználható hasznos információ nyerhető ki. A 3-2. ábrán látható a szegmentálás eredménye.



3-2. ábra – Szegmentálás

A szegmensek közötti átlapolódás 75%-nak lett megválasztva [3], hiszen ezen a ponton elfogadható a redundancia mértéke, és kellőképpen függetlennek tekinthetők a szegmensek egymástól. Átlapolással a kivehető információk mennyisége növelhető, adott hosszúságú adatok esetén.

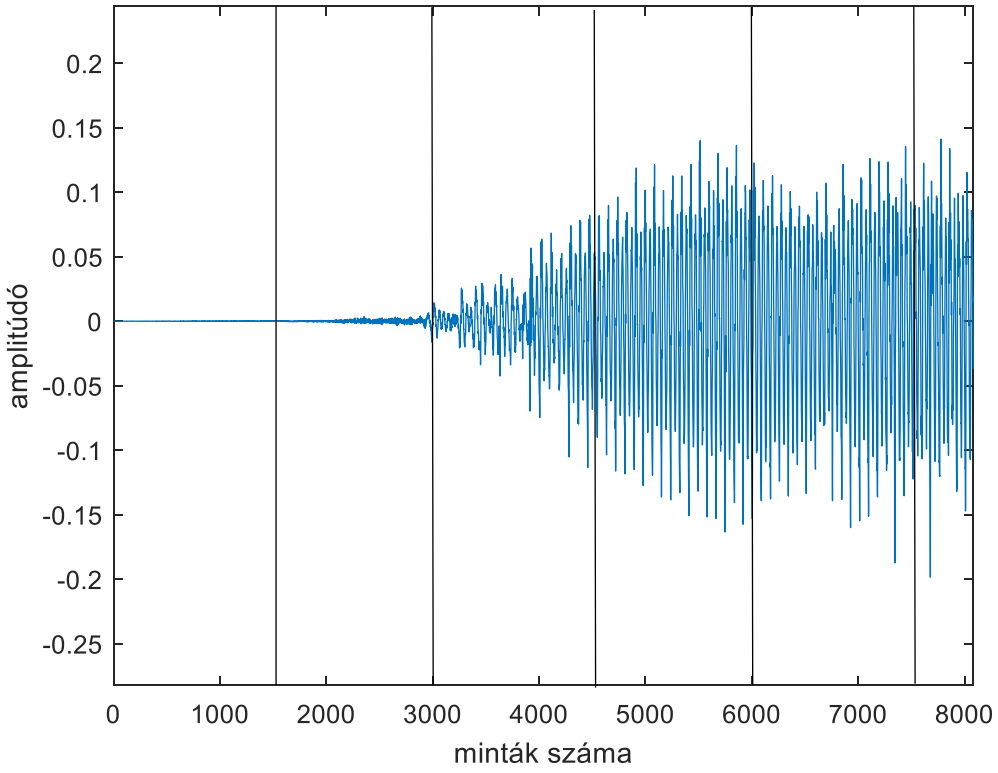
3.2.2. Triggerszint meghatározása

A szegmensek létrehozását követően meg kell határozni, hogy melyek azok a részek, amelyekből később a tulajdonságvektorok generálódnak. Egy szegmens akkor használható fel, ha a háttérzajból jól elkülönülő hangerősségű hangforrást tartalmaz. Ennek vizsgálata a következőképpen zajlik. Minden egyes szegmensre a négyzetes középérték kiszámításra kerül az (1) képlettel, ha egy bizonyos triggerszintet elér annak értéke, akkor az algoritmus felhasználja azt.

Az N darab értéket jelölje $\{x_1, x_2, x_3 \dots x_N\}$. Ekkor ezeknek a számoknak a négyzetes közép értéke:

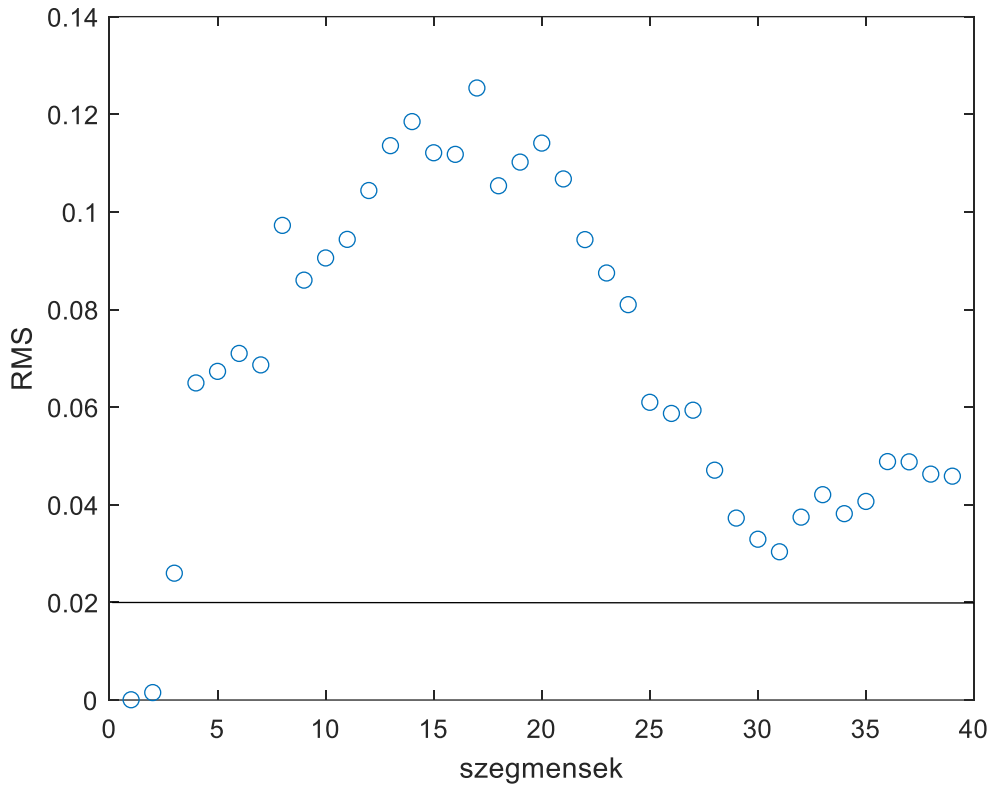
$$x_n = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} = \sqrt{\frac{x_1^2 + x_2^2 + x_3^2 + \cdots + x_N^2}{N}} \qquad (1)$$

A 3-3. ábrán a csecsemő sírás osztály egyik felvételéből látható 181 ms. A függőleges vonalak az egyes szegmensek határait jelölik. 1 szegmens 1500 mintából áll. Észrevehető, hogy a zajszint minimális, a 2. szegmens-nél azonban körülbelül 0.015 nagyságú amplitúdót tapasztalunk.



3-3. ábra - 181 ms hosszúságú csecsemő sírás

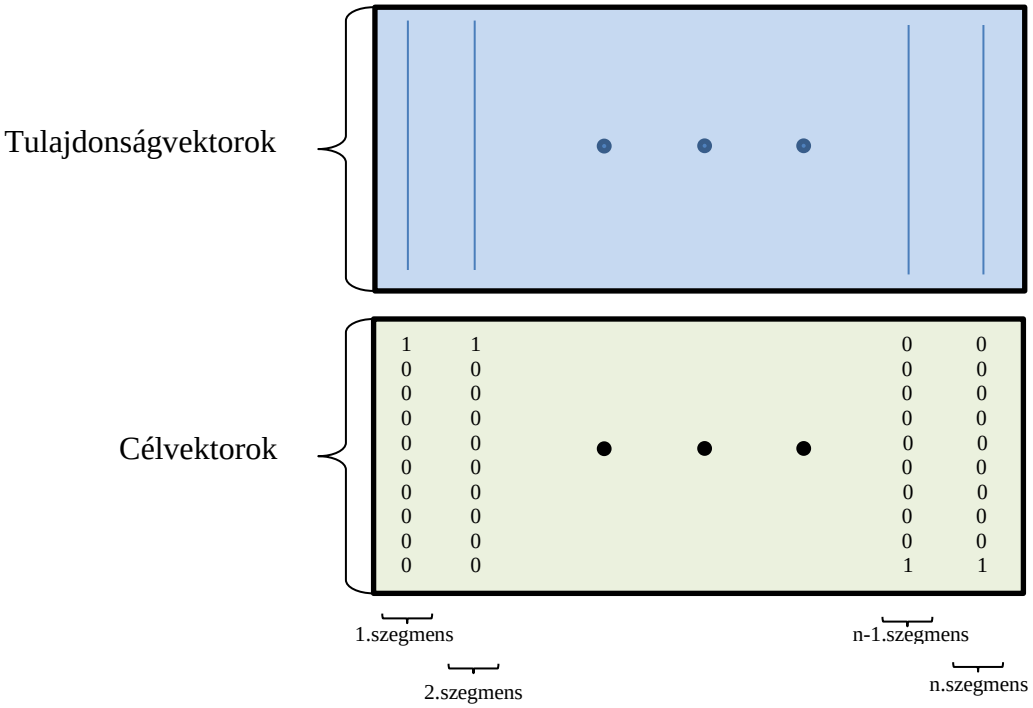
A 3-4. ábrán 40 szegmens RMS értéke látható. A vízszintes vonal jelöli a megválasztott triggerszintet.



3-4. ábra - 40 darab szegmens RMS értéke

3.2.3. Neurális hálózat tanításának rövid áttekintése

A vektorok létrehozásának végeztével azokat fájlokba mentem, hogy a későbbiekben felhasználhassam őket, illetve kiválasztom azokat, amelyek a tanítás során eredményesebbnek bizonyulnak. A 3-5. ábra mutatja, milyen kialakításba rendeződnek az egyes elemek, illetve a hozzájuk tartozó célvektorok (target vectors), amelyek a megfelelő osztályba való tartozást definiálják. One-hot kódolást alkalmazok, ami annyit jelent, hogy a vektor hossza az osztályok számával megegyező, és mindenhol nulla értéket vesz fel, kivéve ott egyest, ahol az aktuális szegmens tulajdonságvektorhoz tartozó osztály szerepel.



3-5. ábra – Vektorok mátrixos szervezése

A következő fázis a neurális hálózat létrehozása és tanítása. Ennek részletesebb kifejtése a 5.2 fejezetben található. A szakdolgozat keretein belül az előrecsatolt neurális hálózatokkal foglalkoztam. Létrehozásuk során az alábbi paramétereket hangolhatjuk:

- Bemenet mérete (Az egyes tulajdonságvektorok hossza határozza meg)
- Rejtett rétegek száma
- Neuronok száma a rejtett rétegekben

A neurális hálózattal kapcsolatos feladatok a Neural Network Toolbox [4] felhasználásával kerültek végrehajtásra. A toolbox lehetőséget nyújt különféle hálózat létrehozására, tanítás folyamatának követésére, illetve tesztelésére. A tanítás egy igen hosszú folyamat, amelyet több tényező is befolyásol pl.: adatbázis mérete, neurális hálózat rejtett rétégeinek száma, neuronok száma. A tanítás gyorsítása érdekében Parallel Computing Toolbox-ot [5] használok. Ennek segítségével párhuzamosítható a tanulás.

A hálózat kimenetének értékelése az ún. confusion mátrix-al történt [7]. A confusion mátrix-ot hiba mátrixnak is szokás nevezni. Klasszifikációs problémák megoldásánál alkalmazzák. A mátrix oszlopai a tényleges osztályt definiálják, míg a sorok a jósolt osztályokat vagy fordítva. A 3-6. ábrán egy példa látható. A mátrix átlójában, amelyet zöld

színnel jelöltem a helyes találatok százalékos eredményei figyelhetők meg. Az egyes oszlopok összege 100%. Ha rossz jóslás történt, tehát például egy macskanyávogásra kutyaugatást kaptam akkor megvizsgáltam, hogy a jósolt eredmények közül hány darab tartozik a kutyaugatáshoz, ezt a számot elosztottam a teljes prognózis számával, így megkaptam a macskanyávogás és kutyaugatás mátrix metszetébe tartozó számot (1.52%). A rossz eredmények pirossal vannak jelölve.

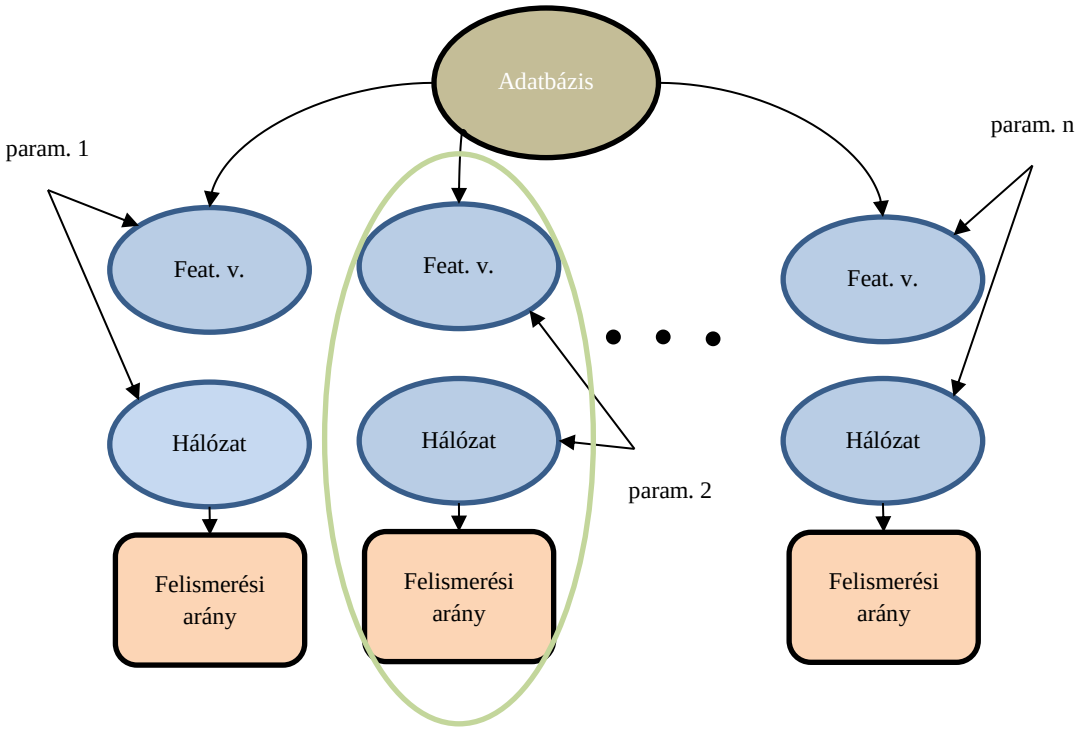
Confusion mátrix	Emberi hang	Kutyaugatás	Macskanyávogás
Emberi hang	99.75%	2.73%	0.2%
Kutyaugatás	0.2%	96.31%	1.52%
Macskanyávogás	0.05%	0.96%	98.28%

3-6. ábra – Confusion mátrix

A kialakított rendszer összes paraméterterének együttes hangolása egy nagyon hosszú folyamatot eredményezett volna, így ettől a megoldástól elhatárolódva, külön a tulajdonságvektorok előállításához szükséges paramétereket hangoltam egy viszonylag jó eredményt szolgáló hálózattal, majd a hálózat paramétereit finomítottam tovább. A különböző paraméterek, különböző felismerési arányt hoznak létre, ezek közül a legnagyobbat kiválasztva meghatározható a rendszer végső paraméterkészlete. A 3-7. ábra ezt a folyamatot illusztrálja.

Az optimális paraméterkészlet meghatározásához össze kell hasonlítani a különböző paraméterezésű hálózatok teljesítményét. Ezért sokszor egyetlen számmal szeretnénk leírni hálózatunk működését, hogy az jól összehasonlítható legyen más rendszerek eredményeivel. A rendszer teljesítményét reprezentáló szám meghatározása azonban nem egy egyszerű feladat, nincs rá egzakt megoldás. A felismerési arány számítása történhet például összes jó jóslás elosztva az összes mintával vagy a confusion mátrix átlójában lévő tagok átlagát is képezhetjük. Az első lehetőség nagy hátránya, hogy a kevesebb mintával

rendelkező osztályok eredménye torzul a több mintával rendelkező osztályok javára. Vegyük például, hogy két osztályunk van. Az egyik osztály 100, míg a másik osztály 1000 mintával rendelkezik. Az első osztály esetében 100-ból 40-re jó válasz érkezett, a második osztálynál pedig 1000-ből 800. Külön szemlélve, az egyik 40%-os felismerési arányt produkált, a második osztály esetében ez az érték 80%. Az átlagukat véve 60%-ot kapunk, ha viszont az összes jó minta osztva az összes mintával, akkor $\frac{840}{1100} = 76.36\%$ az eredmény. Látható, hogy mekkora torzulást okozott az első módszer. Ezt a problémát elkerülve az utóbbi módszer mellett döntöttem, tehát a confusion mátrix átlójában szereplő értékek átlagát vettem.



3-7. ábra – Végső paraméterek kiválasztásának folyamata

3.3. Osztályozás

Az adatbázis 3 részből áll, ahogyan az korábban említésre került 1 tanító és 2 tesztelő halmazból. Az osztályozó algoritmusok feladata, hogy a bemenetként megadott hanganyagból legenerált tulajdonságvektorokat összehasonlítsa a tanításra felhasznált tulajdonságvektorokkal, így azonosítva, hogy mely hangforráshoz tartozhatnak a bemenet mintái. A dolgozatban két algoritmus kerül bemutatásra részletesebben, amelyet az 5. fejezetben találhatunk:

- KNN (k nearest neighbour)
- Neurális hálózat

4. Tulajdonságvektorokat generáló algoritmusok ismertetése

A tulajdonságvektorokat generáló algoritmusok feladata, hogy a kiválasztott szegmensekből a jellegzetesebb tulajdonságokat kiemelje. Ilyen tulajdonság lehet a periodikusság, impulzusszerű változások. Nem csak az időtartománybéli kép adhat számunkra hasznos információkat a jelről, hanem a frekvenciatartományban látható jelleg is. Dolgozatomban főként spektrum alapú tulajdonságvektorokkal dolgoztam.

4.1. Fourier-transzformáció

Jean Baptiste Joseph Fourier (1768-1830) [8] francia matematikus és fizikus a 19. század fordulóján élt és dolgozott ki egy algoritmust, mellyel periodikus és nem periodikus jelek diszkrét pillanatértékeivel kiszámítható a jelet alkotó sinus alapkomponenseinek amplitúdója és frekvenciája.

A Fourier-tétel kimondja, hogy bármely időtartománybéli hullámalakot le lehet írni sinus és cosinus függvények súlyozott összegeként [8]. Ugyanezt a hullámalakot megjeleníthetjük frekvenciatartományban is Egy harmonikus jel időtartományban három információ segítségével teljesen rekonstruálható, ezek az amplitúdó „A”, körfrekvencia „ω_0”, és fázis t=0 időpillanatban „φ(t=0)”. A három közül kettő különösen fontos szerepet tölt be: amplitúdó és körfrekvencia. Frekvenciatartományba áttérve a harmonikus jelet úgy ábrázoljuk, hogy az amplitúdó a körfrekvencia függvénye. Az amplitúdó körfrekvenciától való függését nevezik spektrumnak. A 4-1. ábrán látható egy időtartománybéli, illetve annak frekvenciatartománybéli képe. Az időtartománybéli jel: $x(t) = 1 \cdot \sin(2\pi \cdot f \cdot t)$. A jel amplitúdója egységnyi, a frekvenciája 50 Hz. A körfrekvenciát az alábbi összefüggéssel (2) kapjuk meg:

$$\omega = 2\pi \cdot f \tag{2}$$

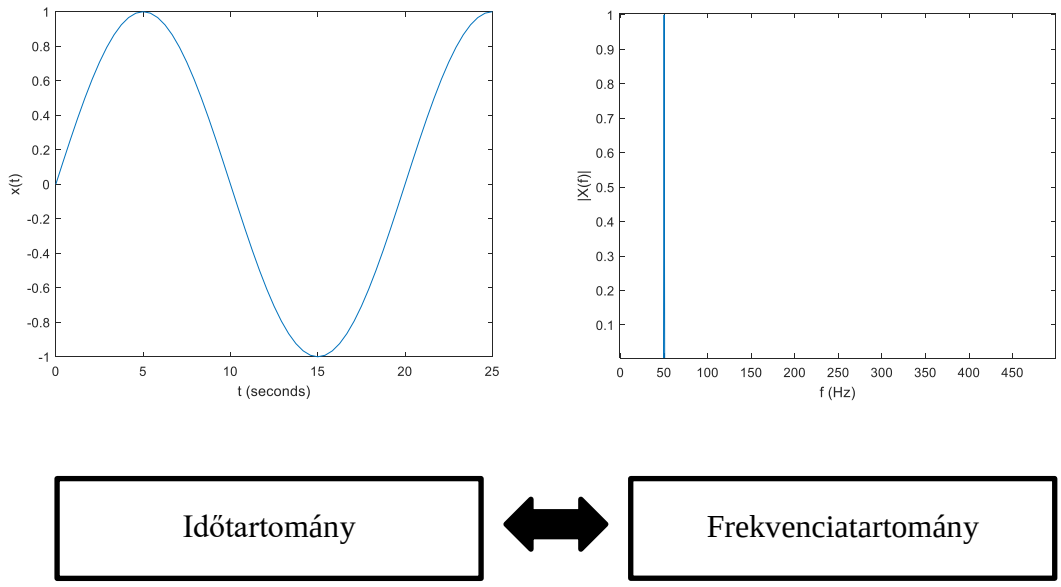
Az időtartományból a frekvenciatartományba való áttérést az (3) egyenlet, míg az inverz műveletet a (4) definiálja [9].

Fourier-transzformáció:

$$X(f) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j2\pi ft}\ dt \tag{3}$$

Inverz Fourier-transzformáció:

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(f) \cdot e^{j2\pi f t} df \tag{4}$$



4-1. ábra - Harmonikus jel idő és frekvenciatartományban [6]

4.1.1. Diszkrét Fourier-transzformáció

A Fourier-transzformáció diszkrét megfelelőjét, amelynek feladata az időtartománybéli mintavételezett jel értékeinek frekvenciatartományba való transzformálása a (5)-os képlet írja le [10].

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\left(\frac{2\cdot\pi\cdot n\cdot k}{N}\right)} \qquad k = 0, 1, 2, \dots, N-1 \tag{5}$$

ahol

$x[n]$ értékek a mintavételezett jel időtartománybéli értékei,

N a minta értékeinek száma.

Ha a jelből egy megadott mintavételi frekvenciával vesszük a mintákat, akkor a (6) képlettel megkaphatjuk a mintavételi időt, vagy is az egyes minták közötti időtartamot.

$$h = \frac{1}{f_s} \tag{6}$$

ahol

h a mintavételi időtartam,

f_s a mintavételi frekvencia.

Frekvenciatartományban is értelmezünk frekvencia felbontást, amelyet a (7) egyenlet ad meg.

$$\Delta f = \frac{f_s}{N} = \frac{1}{N \cdot h} \tag{7}$$

ahol

Δf a frekvencia felbontás,

f_s a mintavételi frekvencia,

N a minták száma,

h a mintavételi idő,

$N \cdot h$ pedig a teljes vizsgálati időtartam.

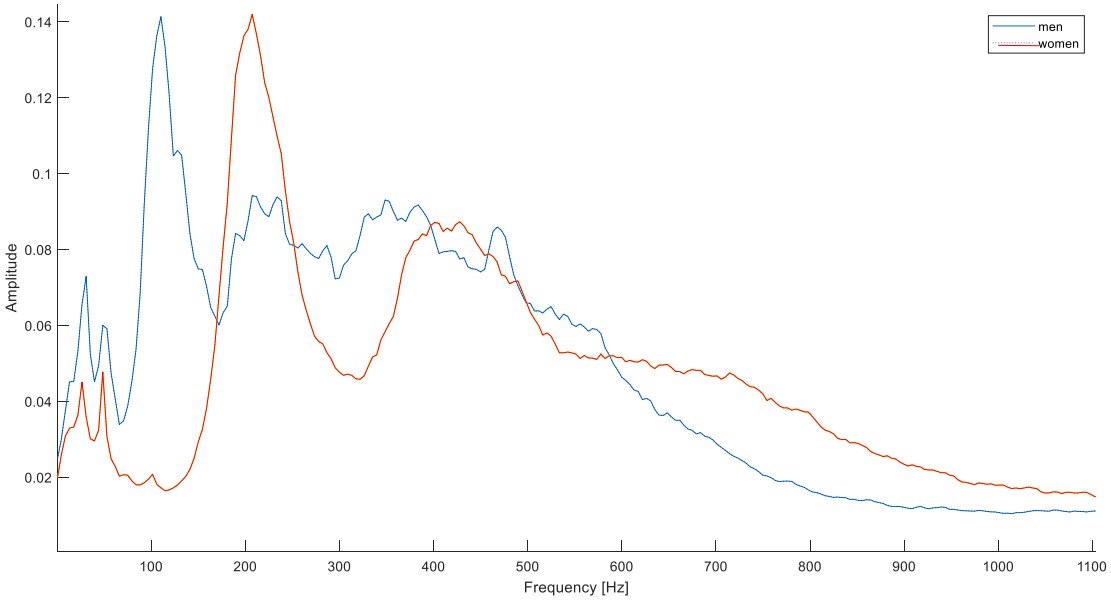
4.1.2. FFT (Fast Fourier Transform)

A diszkrét Fourier-transzformáció egy nagyon időigényes számítási algoritmus. N minta esetén közelítőleg N^2 műveletet kell elvégeznünk. Ezt a problémát orvosolja a gyors Fourier-transzformáció.

Az FFT a diszkrét Fourier-transzformáció kiszámítására szolgál. Az algoritmus műveletigénye nagyságrendileg $N\,logN$ [11]. A mintavételezés frekvenciáját pedig úgy kell megválasztani, hogy legalább kétszer akkora legyen, mint a maximálisan feldolgozandó frekvencia.

A munkám során az egyes hanganyagokhoz tartozó spektrumok előállítására FFT-t használtam. Az 4-2. ábrán példaként az átlagos férfi és női hang látható. Ez az általam felvett 50-50 férfi és női hangból került előállításra FFT algoritmus felhasználásával.

Megfigyelhető, hogy a frekvenciatartománybéli jel milyen plusz információkat hordozhatnak az időtartománybéli jelhez képest. Látható, hogy a főbb frekvenciakomponensek az átlagos férfi spektrumnál (kék színnel jelölve), körülbelül 110 Hz köré csoportosulnak, míg nők esetében ez az érték 207 Hz.



4-2. ábra - Férfi és női átlagos spektrum

4.2. MEL Spektrum

Az FFT egyik legnagyobb hátránya, hogy nincs összehangban azzal a frekvenciaskálával, amelyet az emberi hallásnál megtapasztalhatunk. A természetben előforduló hangok spektruma nem egyenletesen tartalmaz információt, ellenben az FFT által szolgáltatott eredményekkel, amelyek lineáris leképzésűek.

Az emberi fül sokkal jobban érzékeli a kisebb frekvenciákon történő változásokat, mint magasabb frekvenciákon. A fontosabb információk az alacsonyabb frekvenciákon találhatóak meg. A beszéd a halláshoz hasonlóan sem követi a lineáris skálát. Az emberi fül a különböző hangjeleket körülbelül 1000 Hz alatt lineáris, míg 1000 Hz fölött logaritmikus skálán észleli [12]. Gondoljunk bele, hogy 50 Hz és 100 Hz közötti különbség mennyivel észrevehetőbb, mint a 10000 és 10050 közötti. Itt talán meg sem halljuk a különbséget. Felmerül a gondolat, hogy az FFT-vel magasabb frekvenciákon történő számítások feleslegesek számunkra, és csupán plusz zajként jelennek meg ezek a komponensek. Ennek a problémának a leküzdésére használunk Mel-skálát. A Mel a beszédjel

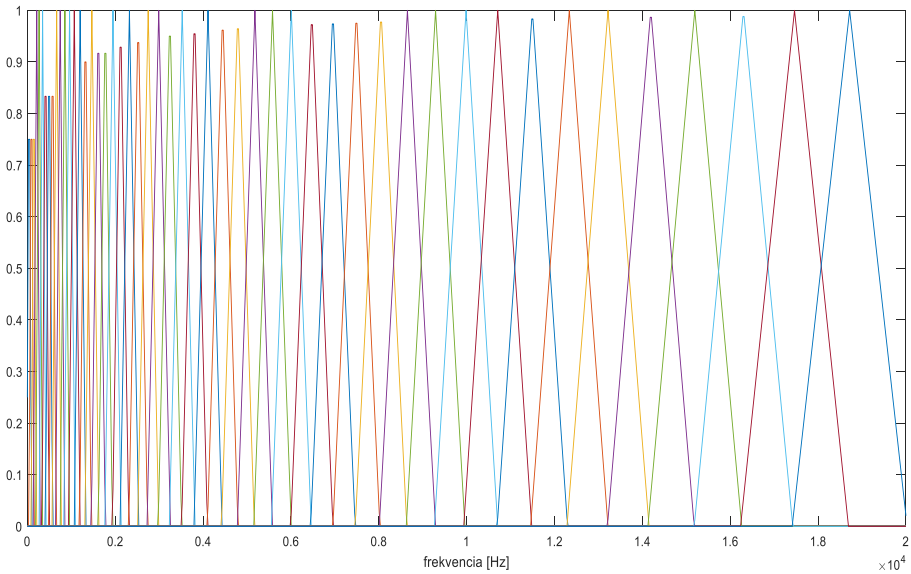
észlelt frekvenciáját társítja a tényleges frekvenciához. Tetszőleges f frekvencia konvertálható Mel-skálába (m) az alábbi (8) összefüggéssel [13]:

$$m(f) = 2595 \cdot \ln\left(1 + \frac{f}{700}\right)$$
$$f(m) = 700 \cdot (10^{\frac{m}{2595}} - 1) \tag{8}$$

Miután a frekvenciákat Mel léptékké alakítottuk át, egy szűrő bankot alkalmazunk, amely háromszög alakú sávszűrőket tartalmaz. Mivel a Mel-skála nem egyenletes eloszlású, ezért a sávszűrők is ezt az elrendezést követik. Kisebb frekvenciákon több szűrő kerül elhelyezésre, míg a nagyobb frekvenciástartományokon kevesebb. Az itt használt háromszög alakú szűrőket a következő (9) egyenletek határozzák meg [13].

$$z_m(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k \le f(m) \\ 1 & k = f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \le k \le f(m+1) \\ 0 & k > f(m-1) \end{cases} \tag{9}$$

A következő 4-3. ábrán a Mel szűrőbank látható. Ezt egy matlabban megtalálható függvénnyel (*melfilter)* valósítottam meg.



4-3. ábra – Matlabban generált Mel szűrőbank

A teljesítmény spektrum kiszámítását a (10), míg a Mel spektrumot az (11) egyenlet írja le [14].

$$S[k] = |X[k]|^2 \tag{10}$$

ahol

$X[k]$ az időtartománybéli jel Diszkrét Fourier-transzformáltja (5)

$$p[l] = \sum_{k=0}^{N/2} S[k] \cdot z_m[k]; \qquad l = 0,1,\dots,L-1 \tag{11}$$

ahol

$S[k]$ a teljesítmény spektrum,

N a Diszkrét Fourier-transzformáció hossza,

L a Mel háromszög szűrők száma,

z_m a Mel szűrőbank.

4.3. Cepstrum

A "cepstrum" elnevezést a "spektrum" első négy betűjének megfordításával hozták létre. Az időtartománybéli jelből kialakított spektrum logaritmusát véve, majd annak az inverz Fourier-transzformáltját a Cepstrum-ot eredményezi. Létezik komplex cepstrum, valós cepstrum, teljesítmény cepstrum, és fázis cepstrum. Jelen dolgozatban csak a teljesítmény cepstrum számítása kerül ismertetésre. Az (12) egyenelet definiálja a teljesítmény cepstrum kiszámítását [15].

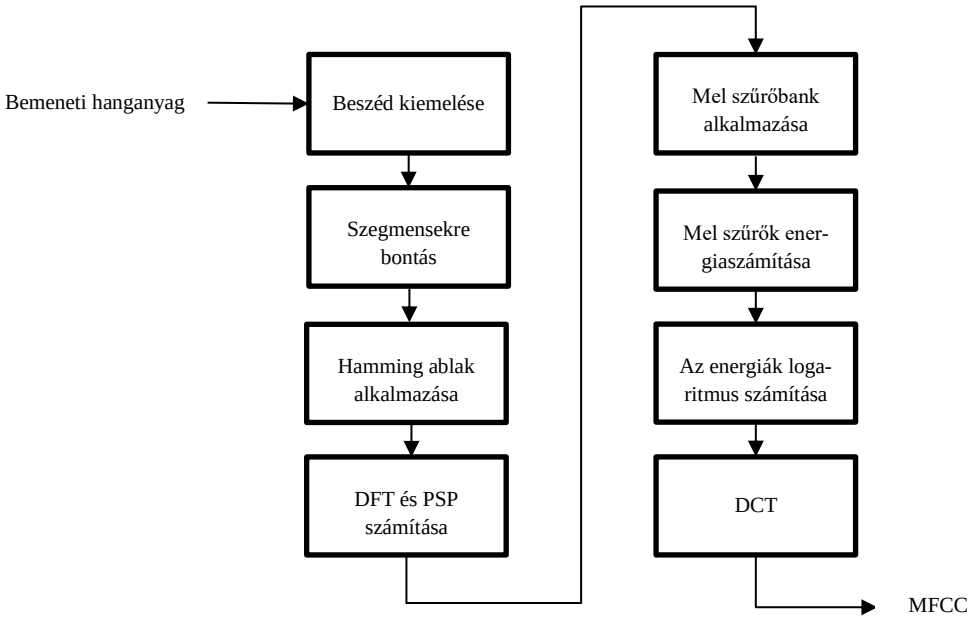
$$Egy\ jel\ teljesítmény\ cepstruma = |F^{-1}\{\log(|F\{f(t)\}|^2)\}|^2 \qquad (12)$$

4.4. MFCC (Mel Frequency Cepstral Coefficients)

Az MFCC egy beszédfelismerésben közkedvelt algoritmus [14]. Az beszédjel egyik legjobb paraméteres reprezentációja. 1980-ban Davis és Mermelstein vezették be [16].

Az MFCC kiszámításának alapvető lépései az alábbiak:

1. A beszédjel kiemelése
2. A jel szegmensekre bontása
3. Hamming ablak alkalmazása a szegmensekre
4. A diszkrét Fourier transzformáció és a Power Spektrum Periodogram becsléseinek kiszámítása
5. Mel szűrőbankok alkalmazása, és a Mel spektrum energiájának megtalálása
6. MFCC meghatározása



4-4. ábra – Az MFCC meghatározásának folyamata [14]

A 4-4. ábrán az MFCC kiszámításának folyamata látható. A következőkben az egyes lépések kerülnek kifejtésre.

4.4.1. Kiemelés

A felvett hanganyag általában erősen terhelt külső zajjal, amely torzítja a rendszer által szolgáltatott eredményeket. A zaj csökkentése érdekében egy elsőrendű felüláteresztő szűrőt használunk, amellyel kapunk egy kiemelt részt $s(n)$ a felvett jelből $x(n)$.

$$s(n) = x(n) - Ax(n-1); \;\; 0 \leq A \leq 1 \tag{13}$$

Az A értéke gyakran 0.95, amely annyit jelent, hogy minden egyes minta 95%-ban származik a korábbi mintákból [14]. Ennek a fő célja az, hogy növelje az energia mennyiségét a magasabb frekvenciákon.

4.4.2. Szegmensekre bontás

A beszédjel egy folyamatosan változó jel, ezért nehéz teljes egészében vizsgálni. Rövid szakaszokon viszont kevésbé intenzív a változás. Ez az oka a beszéd kisebb szegmensekre való bontásának. A szegmensméret egy fontos paraméter, hiszen a kis szegmens kevés mintát, míg a nagy méretű szegmensek sok változó jelet tartalmaznak. Az ideális szegmensméret körülbelül 10-40 ms közé tehető [14]. Minden szegmens rendelkezik N mintával, amelyek közül M átlapolódik a következő szegmenssel.

Természetesen $M < N$, ahol gyakran $N = 256$ és $M = 100$. Ennélfogva a kiemelt jel $s(n)$ használható a következő alakban: $s_i(n)$, ahol n a minták száma az egyes szegmensekben és i a teljes szegmensek száma.

4.4.3. Ablakozás

Ablakozásra Hamming ablakot használ az algoritmus, amely a (14) képlet definiál diszkrét időben.

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right);\ \ 0 \le n \le N-1 \tag{14}$$

Minden egyes szegmens összeszorzásra kerül az imént említett Hamming ablakkal, így kapjuk meg a $s_i(n) \cdot w(n)$ minden egyes i-re.

4.4.4. DFT

A 4.1.1. fejezetben részletesen kifejtésre került a Diszkrét Fourier-transzformáció számítása, ezért itt csak az algoritmushoz szükséges egyenlet kerül felírásra.

$$s_i(k) = \sum_{n=1}^{N} s_i(n) \cdot w(n) \cdot e^{-j\left(\frac{2\cdot\pi\cdot n\cdot k}{N}\right)};\qquad 1 \le k \le K \tag{15}$$

ahol

K a DFT fokának nagysága.

Az energia szintek meghatározása különböző frekvenciákon a (16) egyenlettel történik.

$$p_i(k) = \frac{1}{N} \cdot |s_i(k)|^2 \tag{16}$$

Ezt nevezzük Power Spektrum Periodogram becslésnek.

4.4.5. Mel szűrőbank alkalmazása

A Mel szűrőbankok kialakítása után, amelyről részletesebb információk a 4.2. fejezetben olvashatók, beszorozzuk a (16) Power Spektrum Periodogram becslésével. Ennek eredménye a jel Mel spektruma (17).

$$p_m = \sum_{k=0}^{K/2} p_i(k) \cdot z_m(k); \tag{17}$$

ahol

K a DFT fokának nagysága,

m a szűrő száma,

i szegmens száma.

4.4.6. MFCC meghatározása

A végső lépés az MFCC tulajdonságvektor meghatározása. Ehhez a (17) Mel spektrum logaritmusát majd, hogy frekvenciatartományból visszatérhessünk időtartományba a jel Diszkrét Koszinusz Transzformáltját (DCT) kell venni (18).

$$c_n = \sum_{k=1}^{m} (\log p_k) \cos\left\{n \cdot \left(k - \frac{1}{2}\right)\frac{\pi}{2}\right\} \tag{18}$$

ahol

n a Cepstral együtthatók száma az egyes szegmensekben,

m a szűrők száma az egyes szegmensekben.

4.5. LPC (Linear Predictive Coding)

Linear Prediction egy matematikai művelet, ahol a diszkrét idejű jövőbeni értékeket az előző minták alapján lineáris függvényként becsül.

A leggyakoribb reprezentációja a (19) képlet írja le [17]:

$$\hat{x}(n) = \sum_{i=1}^{p} a_i x(n-1) \tag{19}$$

ahol

$\hat{x}(n)$ a jósolt jel értéke,

$x(n-1)$ az korábban megfigyelt jel érték,

a_i a jósló együtthatói.

A pólusokra teljesülnie kell: p < n feltételnek. A következő (20) egyenlet jósló rendszer átviteli függvényét definiálja [17].

$$\frac{S(z)}{E(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} = \frac{1}{A(z)} \tag{20}$$

A végső cél az a_i együtthatók megtalálása, amely leírja a jósló rendszert. A jóslásnak van hibája, amelynek számítása alább (21) látható:

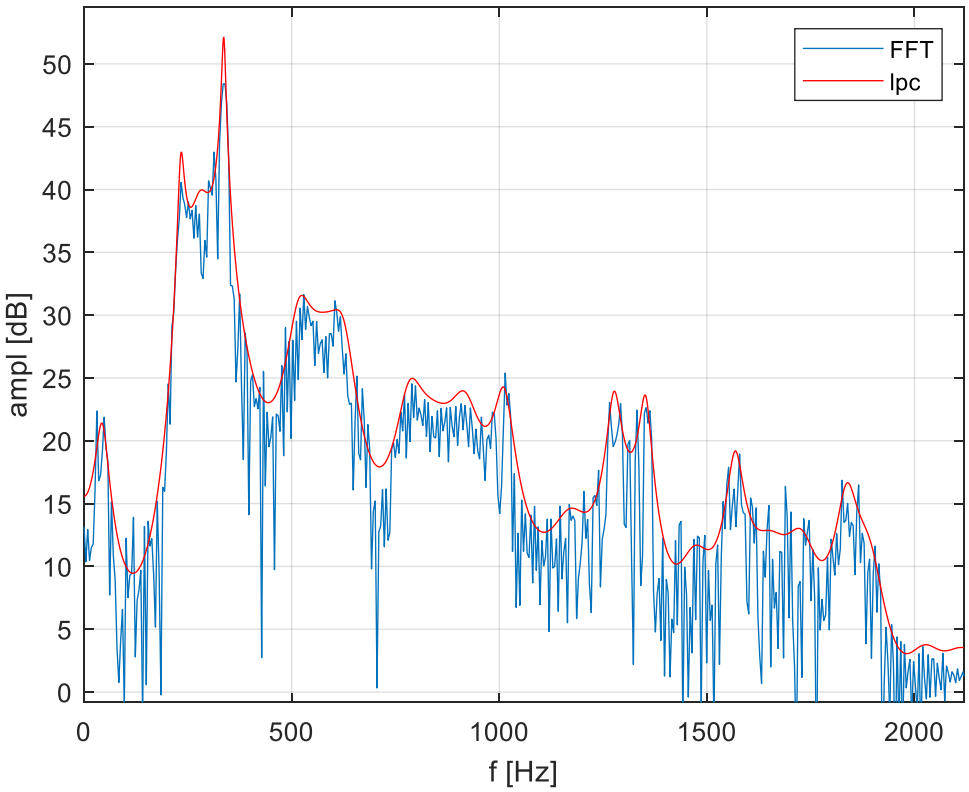
$$e(n) = x(n) - \hat{x}(n) \tag{21}$$

A leggyakrabban választott megoldás az optimális a_i együtthatók megtalálásához az autokorreláció, amely minimalizálja a várható értéke négyzetes hibafüggvényét $E[e^2(n)]$.

A korreláció arra az alapvető kérdésre adja meg a választ, hogy két vagy több változó közötti kapcsolat mennyire szoros. Az autokorreláció pedig egy adott adatsorhoz viszonyítja az adatsor időben eltolt értékeit. A *k*-ad rendű időbeli korreláció formálisan: $r = Corr(x_i, x_{i-n})$. Az időbeli autokorreláció *n*=1 esetén azt jelenti, hogy minden *i*-edik időponthoz tartozó x_i adatot korreláltatjuk az egy időponttal megelőző felvett értékkel [18].

Az autokorreláció elvégzése után egy p hosszúságú vektorban találhatók a kapott együtthatók. Az együtthatók gyorsabb kiszámítására használható még a Levinson-Durbin rekurzió is. A Levinson-Durbin rekurzió megoldásáról részletesebb információ [19] található.

Az 4-5. ábrán egy $p = 50$ -ed fokú LPC által szolgáltatott eredményt láthatunk, amely tökéletes burkolója egy tetszőlegesen választott szegmens spektrumképének. p növelésével a burkoló jobban követi a spektrum változásait. Felhasználásának ötlete, hogy számunkra felesleges információkat hordoz FFT gyorsabb változásai. Elegendő a spektrum jellegét kiemelni. Erre a célra szolgál az LPC.



4-5. ábra - LPC burkolója egy spektrumon

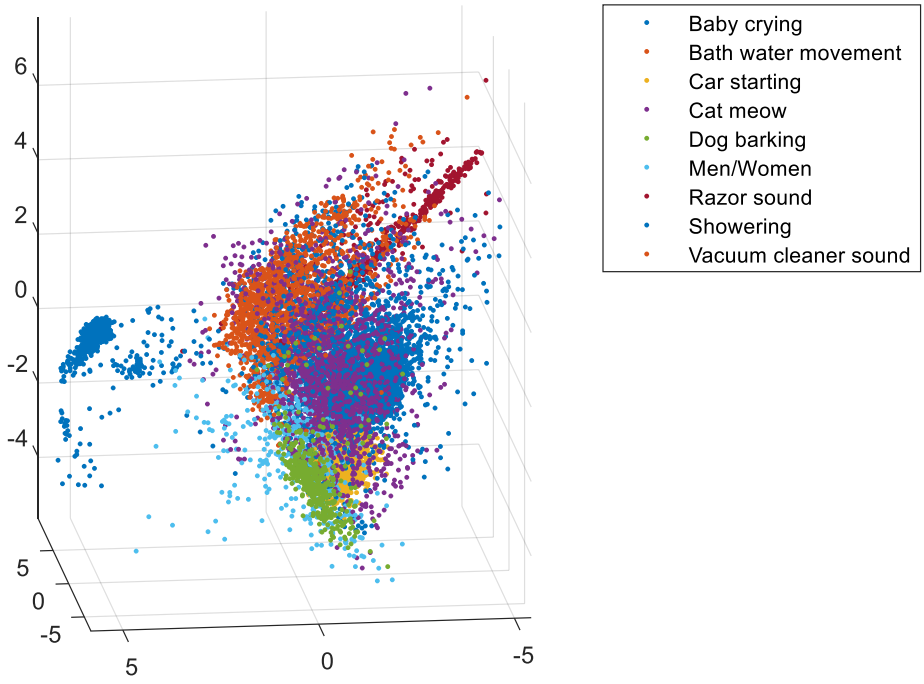
4.6. Reflexiós együtthatók

A reflexiós együtthatók kiszámítását a korábban említett módszerhez hasonlóan korrelációval, majd Levinson-Durbin rekurzióval valósítom meg. Erre a célra szolgál a matlabban is megtalálható függvény [20]:

$[A, E, R] = levinson\ (...)$, ahol A az LPC során kiszámított vektorral egyezik meg, E a jóslás hibafüggvénye, míg R tartalmazza a reflexiós együtthatókat. Paraméterként, ahogy azt az LPC-nél is tettem meg kell adni a p pólusok számát.

5. Osztályozó algoritmusok ismertetése

Az emberi fül számára a kutya ugatása, az emberi beszédtől vagy akár a tusolás hangjától jól elkülöníthető. Erre a tényre alapozva elmondható, hogy az egyes osztályokból legenerált tulajdonságvektorok valamilyen jellegre hasonlóságot mutatnak. Egy L hosszúságú vektor tulajdonképpen egy pontnak feleltethető meg egy L dimenziós térben. Ha a feature vektorokat jól választjuk meg, akkor ezek a pontok várhatóan egymáshoz közel helyezkednek el egy adott osztály esetén. A fenti gondolatmenetet az 5-1. ábra illusztrálja. Az ábrán a már bemutatott adatbázis alapján generált tulajdonságvektorokat ábrázoltam. Mivel grafikusan legfeljebb háromdimenziós alakzatokat tudunk ábrázolni, így az ábrán a PCA (Principal Component Analysis) módszer segítségével kiválasztottam azt a háromdimenziós alteret, amely a legnagyobb információtartalommal bír. Erre a Matlabban elérhető *pca* függvényt használtam. Az ábrán egy adott osztályhoz tartozó pontokat ugyanolyan színnel ábrázoltam, így jól látható, hogy egyazon osztályhoz tartozó pontok viszonylag jól elkülöníthető térrészben találhatók, tehát jól szeparálhatók.

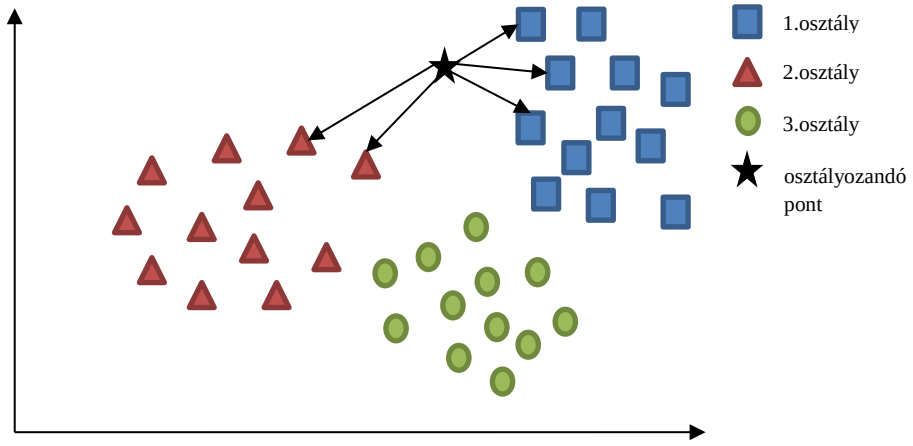


5-1. ábra - 3D-s térben elhelyezett feature vektorok

5.1. KNN (k nearest neighbour)

Osztályozási problémák megoldására mára már rengeteg algoritmus áll rendelkezésre, amelyek különféle módszerekkel képesek az osztályokat egymástól elszeparálni. Ilyenek például a Bayes hálók, Döntési fák, Mesterséges Neurális hálók. Az egyik legismertebb közülük az úgynevezett K-legközelebbi szomszéd algoritmus (KNN) [21].

KNN a nevéből adódóan megvizsgálja azt a *k* vektort, amely a bemenet mintáihoz a legközelebb található. Az egyes tulajdonságvektorokról tudjuk, hogy mely osztályba tartoznak, többségi döntés alapján pedig egyértelműen meghatározható a vektor hovatartozása. Többségi döntés: A *k* kiválasztott vektorhoz tartozó osztályok közül, az az osztály kerül kiválasztásra, amelyre több szavazat érkezett. Az 5-2. ábrán látható példa esetében *k = 5*.



5-2. ábra – A k legközelebbi szomszéd algoritmus

Megvizsgáljuk, hogy az osztályozandó ponthoz, melyik az 5 legközelebbi pont. Az 5-2. ábra alapján három az 1. osztályból, kettő pedig a 2. osztályból került kiválasztásra. A többségi döntés értelmében, így az osztályozandó pont az 1. osztályhoz tartozik.

5.2. Neurális hálózat

A neurális hálózatok olyan számítási feladatok megoldására szolgáló eszközök, amelyek eredete a biológiai rendszerektől származtatható. Az emberi idegsejtek működésének, felépítésének tanulmányozása ihlette azt a fajta elképzelést, hogy kíséreljünk meg bonyolult rendszerek mintájára létrehozni számító rendszereket. Neurális hálózatokat

nagyszámú, egymással összeköttetésben álló hasonló felépítésű neuronok alkotnak, amelyek a legkülönfélébb problémák megoldására bizonyultak alkalmasnak. Néhány ilyen alkalmazási területet említve pénzügyi, gazdasági vagy akár ipari folyamatok előrejelzése, karakterek, kép, vagy egyéb alakzatok felismerése. Főként olyan területeken alkalmazott, ahol nem ismert jelenleg az algoritmikus megoldás, vagy ha az mégis létezik, olyannyira sok számítási műveletet igényel, hogy az reális időn belül a mostani technológiának megfelelő legnagyobb számítógépekkel sem oldható meg.

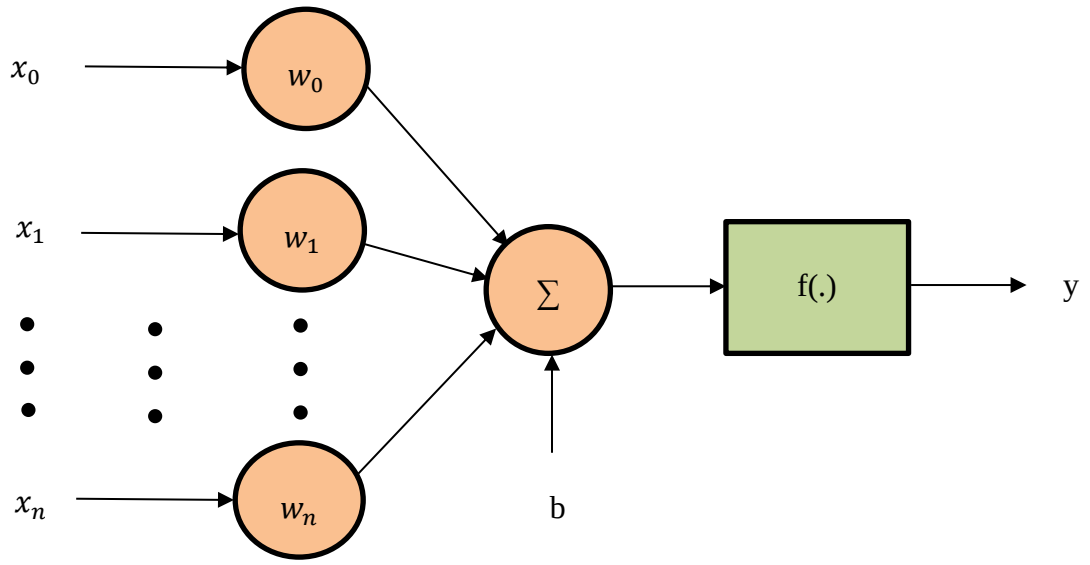
A gyakorlati alkalmazások körét még jobban szélesítő tulajdonság a neurális hálózat nagy mértékű párhuzamossága, amely nagysebességű és robusztus működést, egyfajta hibatűrő képességet biztosít. Köztudott, hogy bizonyos balesetek során, ha az emberi agy egy adott része megsérül, és elveszti annak funkcionalitását, nem feltétlenül jelenti bizonyos képességek elvesztését. Idővel más agyterületei át vehetik a sérült agyrész feladatait. Másik fontos tulajdonság az adaptációs képesség, a környezet változásához való alkalmazkodás, amelyet a folyamatos tanulás képessége tesz lehetővé.

Neurális hálózatok rendelkeznek tanulási, illetve előhívási algoritmussal. Tanulási algoritmusnál általában a minta alapján való tanulást értjük. Előhívási algoritmus pedig a megtanult információk felhasználásáért felelős.

A neurális hálózatok működésénél tipikusan két fázist különböztethetünk meg. Az első fázis a tanulás fázis, ahol a tanításra szánt mintákban rejlő információkat valamilyen formában eltároljuk. Lassú, hosszú folyamat, amely esetenként sikertelen tanulási szakaszokat is tartalmaz. Ez a rész a hálózat paraméterének behangolását szolgálja. Felhasználása a második fázisban az előhívási fázisban kerül sor, amely egy jelentősen gyorsabb információs feldolgozást jelent [22].

5.2.1. Neuronok felépítése

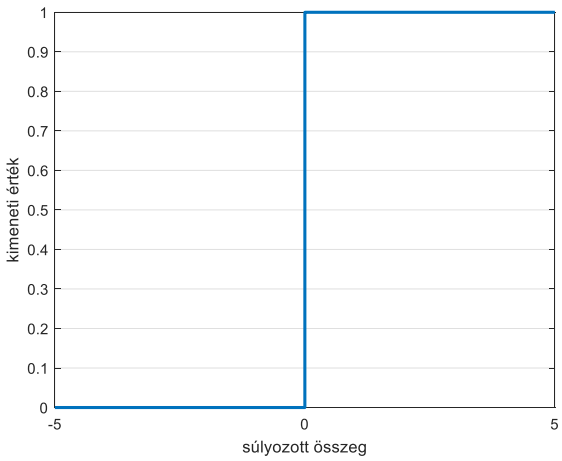
A neuron egy több bemenetű, egy kimenetű struktúra, amely általában a bemenetek és a kimenetek között egy nem lineáris leképzést biztosít. Egyes neuronok képesek korábbi állapotinformációk tárolására, rendelkeznek memóriával. A bemeneti és a tárolt információkból tipikusan egy nem lineáris függvény felhasználásával képezi a kimenet értékét. Ezt a függvényt aktivációs függvénynek nevezzük. (Szokás transzfer függvénynek is hívni).



5-3. ábra - Neuron felépítése

A fenti 5-3. ábrán egy memória nélküli neuron, vagyis perceptron látható. A perceptront 1957-ben Frank Rosenblott találta fel [23]. Ennek segítségével a tanítás befejeztével képes két lineárisan szeparálható bemeneti mintahalmazt elválasztani egymástól. A lineáris szeparáltság azt jelenti, hogy a bemeneti mintateret egy síkkal két diszjunkt tartományra tudjuk szét bontani úgy, hogy a szétválasztott két tartomány eltérő osztályba tartozó mintapontokat tartalmazzon.

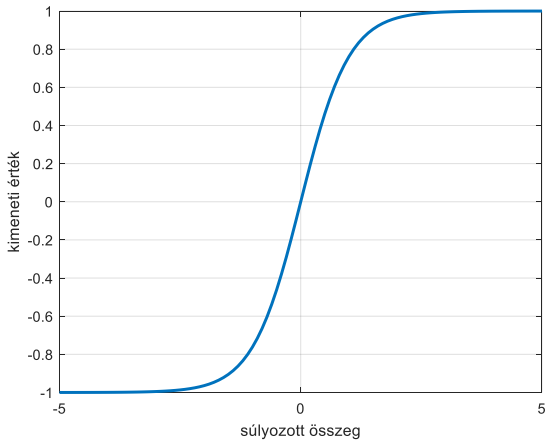
x_0, x_1, … x_n a perceptron bináris bemenetei, amelyek a w_0, w_1, … w_n súlyozással kerülnek összegzésre, majd a súlyozott összeg egy küszöbfüggvényre kerül. A küszöbbfüggvény értéke 1, vagy 0 lehet. Ez a 5-4. ábrán látható.



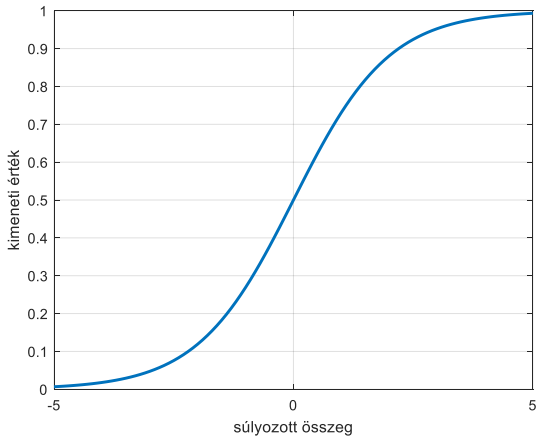
5-4. ábra – Küszöbfüggvény

Ha a súlyozott összeg elér egy bizonyos *b* küszöbértéket, amelyet bias-nek szokás nevezni, akkor a perceptron aktivizálódik. A súlyozott összeget szokás ingernek, míg a kimeneti jelet válasznak nevezni.

Ezzel szemben a neuron bemenetei már nem csak bináris, hanem skalár értékeket is felvehetnek. A küszöbfüggvény helyére pedig más nemlineáris elemek is kerülhetnek, amelyeket a lentebbi 5-5. és 5-6. ábrákon láthatunk [24].



5-5. ábra - tanh függvény



5-6. ábra - sigmoid függvény

A neuron kimenete az alábbi egyenlettel (22) határozható meg:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{22}$$

ahol

- w_i a neuron egyes súlyai
- x_i a neuron egyes bemeneti értékei
- b a neuron bias értéke
- f a neuron aktivációs függvénye
- y a neuron kimenete

5.2.2. A neurális hálózatok topológiája

A hálózat topológiáját egy irányított gráfnak feleltethetjük meg, ahol a gráf csomópontjai a neuronok, míg a kapcsolatot a neuronok, a kimenet, és bemenet között a gráf élei reprezentálják. Az éleket a bemenettől a kimenet felé irányítjuk.

A gráf csomópontjai nem feltétlenül vannak egymással kapcsolatban, így lehetőség van, hogy a gráf csomópontjainak halmazát diszjunkt részhalmazokra bontsuk. Ezek alapján 3 neuron típust különböztetünk meg [22].

- Bemeneti neuronok: egy bemenetű, egy kimenetű neuronok, amelynek bemenete a hálózat bemenete, kimenetük pedig más neuronok meghajtására szolgál.

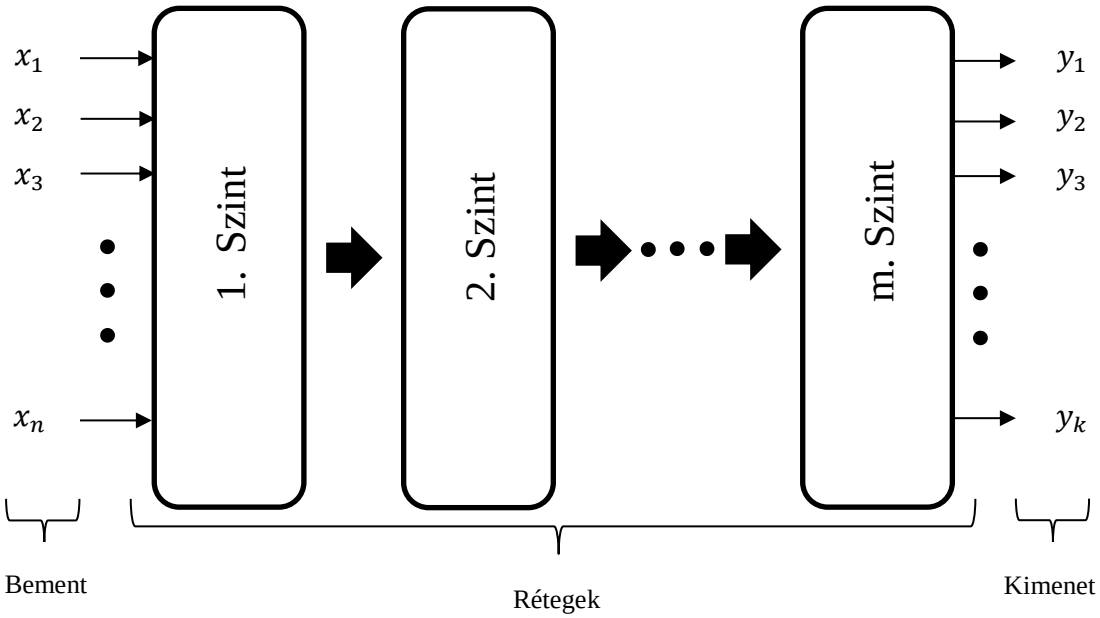
- Kimeneti neuronok: bemenete más neuronok kimenete, kimenete a rendszer válaszát szolgáltatja.
- Rejtett neuronok: bemeneteik és kimeneteik kizárólag csak más neuronokhoz kapcsolódnak.

A neuronokat általában rétegekbe (layers) szervezzük. Egy rétegbe hasonló típusú neuronok szerepelnek. A neuronok közötti kapcsolat is megegyező rétegen belül. Bemeneti rétegről (input layer) akkor beszélünk, ha a rétegbe tartozó neuronok bemenetei a teljes hálózat bemenetei. A bemeneti neuronok buffer jellegűek, információfeldolgozást nem végeznek csupán a következő réteg bemeneteinek kiszolgálása a feladata. A rejtett réteg (hidden layer) esetében a neuronok, más réteg neuronjainak kimeneteihez kapcsolódnak, kimeneteik pedig szintén más rétegbéli neuronok bemeneteit képezik. Kimeneti réteg a teljes hálózat kimenetét képezi. Ennek megfelelően egy rétegbe legalább egy bemeneti rétegnek és kimeneti rétegnek szerepelnie kell. Közöttük tetszőleges számú rejtett réteg helyezkedhet el.

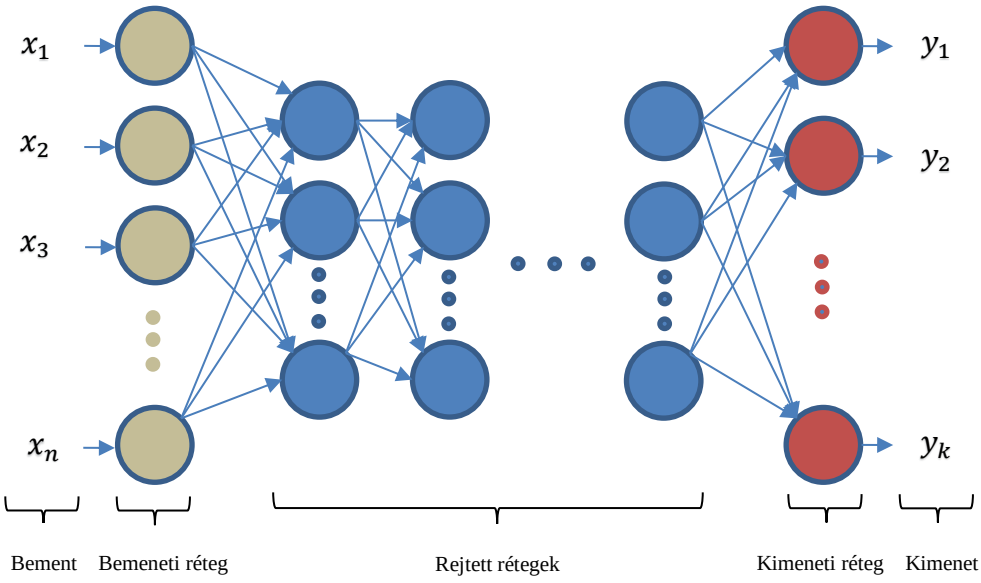
A neuronhálókat a neuronok közötti összeköttetések alapján két fő csoportba sorolhatjuk:

- Előrecsatolt hálózat (Feedforward network)
- Visszacsatolt hálózat (Recurrent network)

Akkor beszélünk visszacsatolt hálózatról, ha a háló topológiáját reprezentáló irányított gráf hurkot tartalmaz, más esetben a hálózat előrecsatolt. Egy előrecsatolt hálózatra mutat példát a 5-7. és 5-8. ábra.



5-7. ábra – Neurális hálózat felépítése



5-8. ábra – Előrecsatolt neurális hálózat

5.2.3. A neurális hálózat tanítása

A neurális hálózatok egyik legfőbb képesége a tanulási képesség. Viselkedésüket a környezetükből tapasztaltak alapján képesek változtatni. A viselkedés módosítása általában arra irányul, hogy az előzetesen beadott mintákra a hálózat a kívánt válaszokat eredményezzen. De az is előfordulhat, hogy nem ismert az elvárt válasz, és a hálózatnak a feladata a bementekben valamilyen szabályosságot, hasonlóságot vagy különbséget találni. Megkülönböztetünk olyan esetet is, amikor a kívánt választ szintén nem ismerjük, csak annyit tudunk, hogy a rendszer által eredményezett válasz helyes, vagy nem helyes. Az adaptív rendszerek fő jellemzője, hogy nem rögzített képességekkel rendelkeznek, nem egy konkrét feladatot látnak el, képesek a környezetükhöz alkalmazkodni.

A neurális hálózatok tanítása során két fő típussal találkozhatunk [25]:

Ellenőrzött tanulás (Supervised learning): Ennél a tanítási módszernél ismert mind a bemenet, mind a kimenet, úgynevezett pontpárokat adunk a rendszernek. A hálózat feladata pedig, hogy megtanulja a pontpárok által reprezentált bemenet és kimenet közötti leképezést. Ezek felhasználásával hangolódnak a hálózatban jelenlévő súlyok. A hálózat által meghatározott kimenet, és a tanító bemenet különbségének minimalizálására törekszik. Abban az esetben, amikor a visszacsatolt információ mindösszesen csak egy bitnyi, amely azt árulja el, hogy szükséges-e módosítani a hálózatot vagy sem. Viszont annak mértékéről semmilyen információnk nincs. Az ellenőrzött tanulásnak ezt a fajtáját megerősítéses tanításnak nevezzük.

Nemellenőrzött tanulás (Unsupervised learning) Az önálló tanulás fő jellemzője, hogy nem áll rendelkezésünkre a kívánt válasz, emiatt nem lehet cél a meghatározott be- és kimeneti leképzés. A hálózat feladata a bemenetként megadott adatok közötti összefüggések, kapcsolatok megtalálása. A környezetből semmifajta visszajelzés nem érkezik, amely a hálózat helyes működésére utalna. Főként adatelemzésre, statisztikai feladatok megoldására alkalmazzák ezt a tanítási módszert.

Munkám során ellenőrzött tanulást használtam, így a következőkben ez kerül bemutatásra. A tanítás előtt definiálni kell a kívánt válaszokat a hálózat számára. Erre a célra a one-hot kódolást alkalmaztam. A rendszer összesen kilenc osztállyal dolgozik. $t(x) = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0]^T$, $t(x)$ a hálózat által elvárt egyik kimenetét mutatja, ahol az 1-es azt az osztályt jelöli, amelyik osztály mintáit adjuk a rendszer bemenetére. A T pedig a

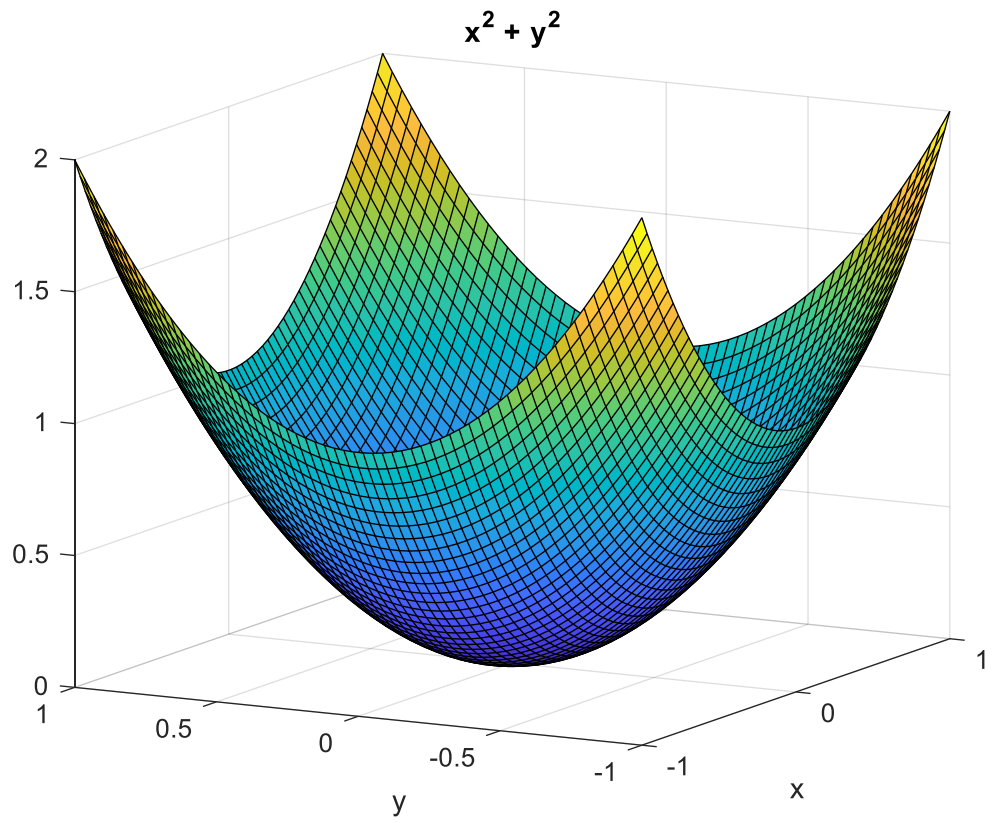
transzponálás jele, hogy ne sor, hanem oszlopvektorokkal dolgozzak. A cél a hibafüggvény minimalizálása, amelyet (23) egyenlet definiál.

$$C(w,b) \equiv \frac{1}{2n}\sum_{x}\|t(x)-a\|^2 \tag{23}$$

ahol

- w a hálózat összesített súlyát jelzi,
- b a hálózat bias értékeit,
- n a hálózat bemenetének száma,
- a hálózat kimenetének eredménye (vektor), amikor a bemenet x.

A $C(w,b)$ értéke annál kisebb minél közelebb van a hálózat kimeneti érétke (a), az elvárt $t(x)$-hez. Az a függ x, w és b-től. Tehát a költségfüggvény hangolását ezeknek a paramétereknek az állításával tehetjük meg. Példaként figyeljük meg 5-9. ábrát. Tekintsük ezt a háromdimenziós alakzatot egy költségfüggvénynek. A háromdimenziós felület paraméterei *x* és *y*, melyek megfeleltethetők w-nek és b-nek. A felületen x és y változtatásával mozoghatunk. Létezik egy olyan x és y érték, amelynél a költségfüggvény minimumot vesz fel. Ezen a ponton hasonlít a legjobban a kimeneti vektor, a $t(x)$ target vektorra. A súlyok és bias-ok változtatásához meg kell határozni $\nabla C(x,y) = (\frac{\partial C(x,y)}{\partial x}, \frac{\partial C(x,y)}{\partial y})$.



5-9. ábra – Költségfüggvény

∇C megadja, hogy x és y változtatásával, milyen mértékben változik C. A kiszámított ∇C-vel hangoljuk paramétereinket a következő képpen [25]:

Legyen Δx az x irányba való elmozdulás, míg Δy az y irányába. Az együttes elmozdulást definiálja $\Delta xy = (\Delta x, \Delta y)$. Δxy-t (24) képpen számítsuk ki, felhasználva ∇C-t.

$$\Delta xy = -\eta \nabla C \tag{24}$$

ahol

η tanulási ráta, amely a minimum elérésének gyorsaságát szabályozza.

Minél nagyobb az értéke, annál gyorsabban éri el $C(x, y)$ a minimumot. Fennáll annak a veszélye, hogy kellően nagy változások esetén a minimumot „átugorva” sosem jutunk el oda. Túl kis érték mellett, pedig nagyon kis változások történnek, ami miatt a minimum keresés folyamata nagyon sok időt emészt fel.

x és y új értékének meghatározása alább látható:

$$x \to x' = x - \eta \nabla C$$
$$y \to y' = y - \eta \nabla C \tag{25}$$

Ezzel a módszerrel megtalálható x és y azon értékekei, amelyekkel $C(x, y)$ minimum pontjában lehetünk.

Természetesen létezik több olyan tanító algoritmus, amely sokkal hatékonyabb és gyorsabb megoldást biztosít a minimum megtalálására. Néhányat megemlítve közülük:

- **Gradient descent backpropagation**: a súlyok változtatása a legmeredekebb irányokban történnek, fix tanulási ráta mellett. Egyik hátránya, hogy könnyen beragadhat lokális minimumba, a másik hátrány, hogy fix tanulási rátának köszönhetően lassan konvergálhat, és akár oszcillálhat is az algoritmus.
- **Conjugate gradiant backpropagation:** a keresések konjugált irányokban történnek, hogy meghatározhassuk azt a tanulási rátát, amely mellett a hiba minimális. Ez a módszer gyorsabb konvergenciát eredményez, viszont a konjugált irányok számítása időigényesebb.
- **Levenberg-Marquard backpropagation:** Sok memóriát igényel, de a tanítási algoritmusok közül a leggyorsabb.

6. Teszteredmények

Az előző fejezetekből láthatók, hogy a kialakított rendszer számos paraméterrel rendelkezik, amelyek kulcsfontosságú szerepet töltenek be a működés szempontjából. Ezek megfelelő megválasztása egy hosszadalmas folyamat. A munkám nagyrészét ezek kialakításával töltöttem.

6.1. Paraméterek hangolása

A paraméterek két fő halmazra oszthatók. Az egyik a tulajdonságvektorok generálásához szükséges, míg a másik a neurális hálózat megfelelő kialakításáért felel.

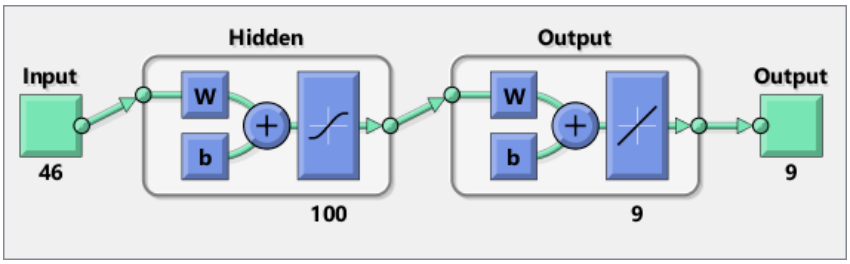
1. Tulajdonságvektorok generálásához szükséges paraméterek:

- Szegmens hossz
- Átlapolódás mértéke
- Triggerszint
- MEL háromszög szűrők száma
- LPC, Reflexiós együtthatók fokszáma

2. Neurális hálózat paraméterei:

- Tanító függvény
- Tulajdonságvektorok fúziója
- Rejtett rétegek száma
- Rejtett rétegekben lévő neuronok száma
- Rejtett rétegek aktivációs függvényei
- Elfogadási határ (Utófeldolgozó blokk)

Első lépésként egy olyan hálózat kialakítása volt a cél, amely jó alapként szolgálhat a vektorok paramétereinek megtaláláshoz. Fontos szempont volt a gyors taníthatóság, így egyszerűbb felépítésű hálózatokban gondolkodtam. Az első kialakított hálózat az 6-1. ábrán látható.

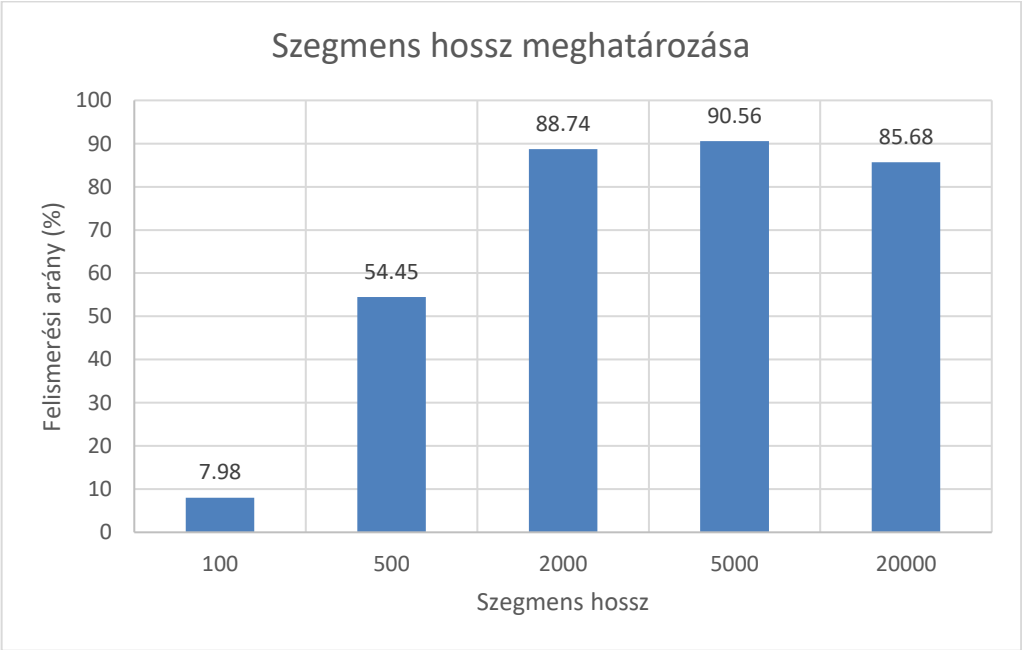


6-1. ábra – Egyszerű neurális hálózat

Az 6-1. ábrán az egyes rétegek alatt a neuronok száma található. A rejtett és kimeneti réteg aktivációs függvényei előre definiálhatók, a rejtett rétegnél tanh-t, míg a kimeneti rétegnél lineáris leképzést használtam. Az *input* méretet a tulajdonságvektorok hossza határozza meg, míg a *kimenetét* az osztályok száma. A tanító algoritmusok közül a matlabban elérhető *trainlm*-re esett a választás, amely a Levenberg-Marquard backpropagation elven működik. Ennek oka a gyors algoritmikus működés.

6.1.1. Szegmens hossz

A szegmens hosszúságának meghatározása során egy viszonylag nagy értéktartományban kerestem. A tartomány két határa a 100 és 50000 minta/szegmens. A köztük lévő részeket pedig úgy osztottam fel, hogy nagyságrendileg megmutatkozzon a helyes beállítandó érték. A többi paraméter megválasztását néhány korábbi kísérlet előzte meg, ezek alapján kerültek beállításra. Az eredmények a 6-2. ábrán láthatók.

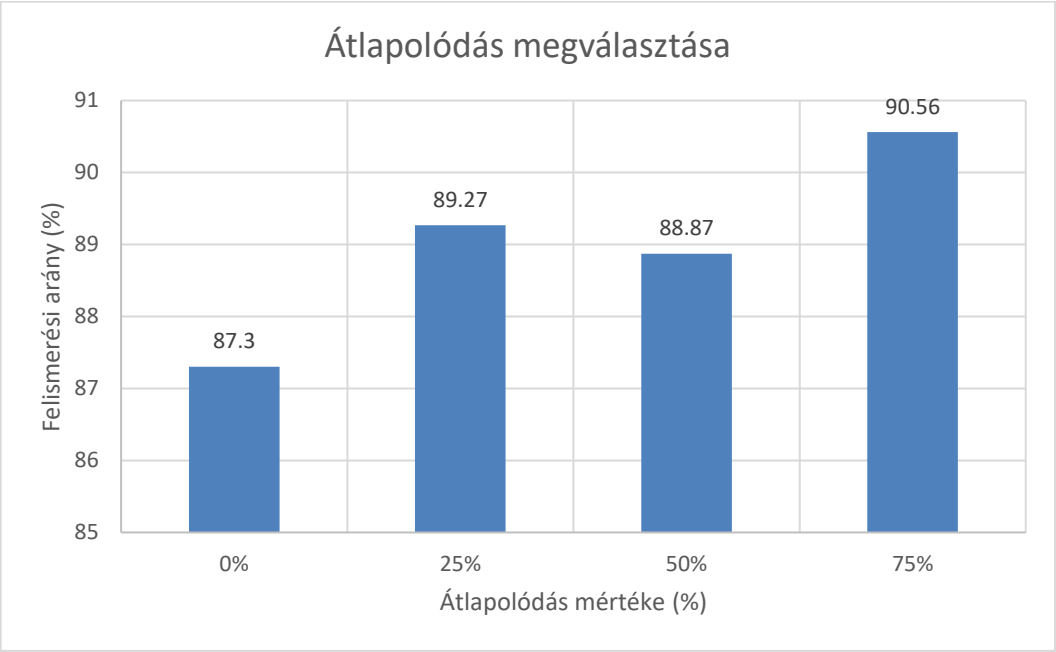


6-2. ábra – Szegmens hossz meghatározása

Elmondható, hogy 2000 és 20000 közötti szegmens méret megválasztása 85% feletti felismerési arányt eredményez. 500 minta/szegmens érték alatt viszont, olyannyira rövid lesz az egy szegmensben lévő hanganyag hossza, hogy a rendszer már nem képes megfelelő döntést meghozni annak hovatartozásáról. Ezen információk alapján a munkám hátralévő részében 5000 mintát tartalmazó szegmensekkel dolgoztam.

6.1.2. Átlapolódás mértéke

A szegmensek méretének meghatározása után a következő lépés a köztük lévő átlapolódás mértékének megválasztása. Az egyes értékeken 25%-os lépés közökkel lépkedve figyeltem meg a rendszer teljesítményét.



6-3. ábra. Szegmensek közötti átlapolódás megválasztása

Az átlapolódás mértékének növelésével egyre több mintán tudok elvégezni műveleteket. Ez többlet információval járhat a rendszer számára, amelynek eredménye a 6-3. ábrán jól látható. A 0%-ról a 75%-ra való váltás körülbelül 3%-os javulást eredményezett a felismerési arányban. A továbbiakban 75%-ra választottam az átlapolódás nagyságát.

6.1.3. Triggerszint megválasztása

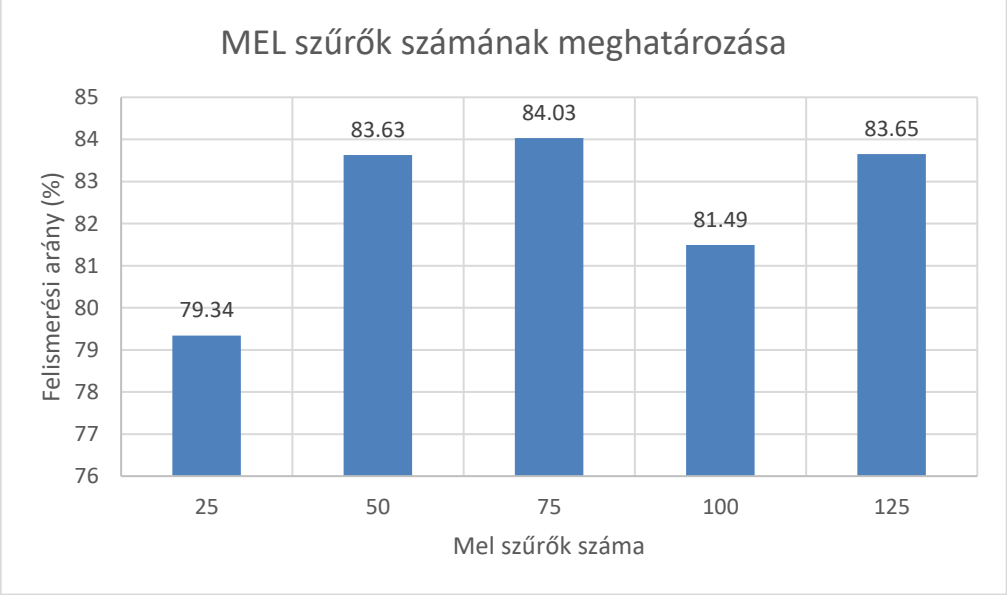
A hangminták rögzítése során sok olyan környezeti tényező játszhat közre, amely zajként jelenik meg a felvételben. Annak érdekében, hogy ezt a zavaró hatást minél jobban elnyomjuk szükséges egy olyan szintet megválasztani, amelyről tudjuk, hogy az alatt semmilyen hasznos információ nem szerepel, csak is a környezetből beszűrődő zaj. A szegmensek kiválasztása során azokat, amelyek nagyrészben csak zajt tartalmaznak figyelmen kívül maradnak, ezzel javítva a rendszer eredményeit. Ennek a szintnek a megválasztása a következő képpen történt. Megfigyeltem egy időtartománybéli jelnek (3-3. ábra), mely az a pontja, ahol a hasznos jel megjelenik. Kiszámoltam annak a szegmensnek az RMS értékét, amelybe a hasznos jel kezdete tartozik, majd ezt a határt megválasztva elkülönítettem a zajt és a rendszer számára felhasználható jelet (3-4. ábra). A következő 6-1. táblázat tartalmazza, hogy milyen a rendszer felismerési aránya triggerszinttel, illetve anélkül. (A zölddel jelzett cellák a nagyobb százalékos arányt, míg a bordó a kisebbet jelölik.) Látható, hogy triggerszint alatti teljesítménnyel rendelkező szegmensek elhagyása nagyban javítja a felismerési arányt.

Felismerési arány	Triggerszint nélkül (%)	Triggerszinttel (%)
Csecsemő sírás	80.34411	85.06629449
Fürdés közben keltett mozgások hangja	87.4562	90.7120743
Autó indítás	98.27651	100
Macskanyávogása	58.02418	72.43667069
Kutyaugatása	44.98715	85.02202643
Férfi/Női beszéd	87.71552	89.34240363
Borotválkozás	93.88084	98.24561404
Tusolás	86.28594	88.25048418
Porszívózás	78.56135	90.77212806

6-1. táblázat – Felismerési arány triggerszinttel és anélkül

6.1.4. MEL háromszög szűrő számainak meghatározása

A szűrők feladata, hogy a lineáris skálából egy logaritmikust állítson elő. A szűrők növelésével a nagyobb frekvenciákon lévő komponensek is nagyobb hangsúllyal szerepelnek az információfeldolgozásban. A megfelelő szűrő szám megtalálásához kiválasztottam egy tartományt (25, 125), és megfigyeltem, hogy ezen a tartományon a különböző beállításokkal a rendszer milyen eredményekkel szolgál. Ezt a 6-4. ábra mutatja. Az eredmények alapján azt a következtetést lehet levonni, hogy 50 darab szűrő használata felett a felismerési arány 80%-ot meghaladja. A legjobb teljesítményt eredményező értéket kiválasztva a továbbiakban a szűrőszámot 75-nek választom meg. A 6-2. táblázaton a rendszer teljesítménye látható osztályonként FFT és Mel-spektrum esetében. Azt mutatja meg, hogy a hálózat számára a lényegesebb információk főként az alacsonyabb frekvenciákon jelennek meg. Többször a Mel-spektrummal jobb eredményeket kapunk, mint FFT-vel.



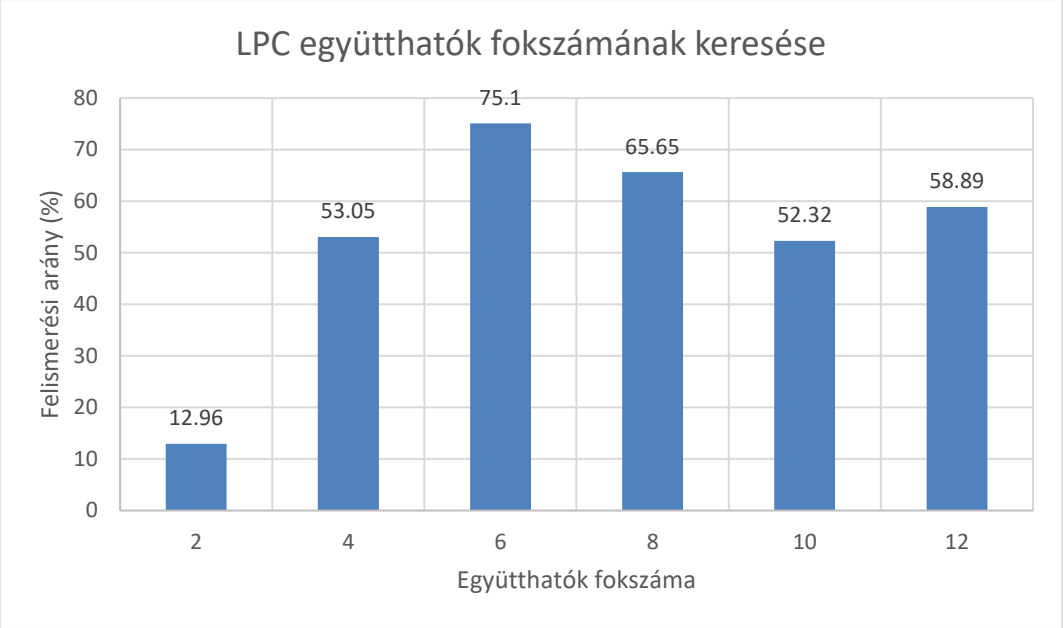
6-4. ábra – Mel szűrők számának meghatározása

Felismerési arány	FFT (%)	Mel Spektrum (%)
Csecsemő sírás	71.52826	77.0412
Fürdés közben keltett mozgások hangja	93.60165	88.1321
Autó indítás	100	100
Macskanyávogása	56.2123	59.5296
Kutyaugatása	84.80176	79.0749
Férfi/Női beszéd	50.79365	85.941
Borotválkozás	95.26316	98.0702
Tusolás	64.45018	77.9212
Porszívózás	73.63465	90.5838

6-2. táblázat – FFT és Mel Spektrum eredményei

6.1.5. LPC, Reflexiós együtthatók fokszámainak meghatározása

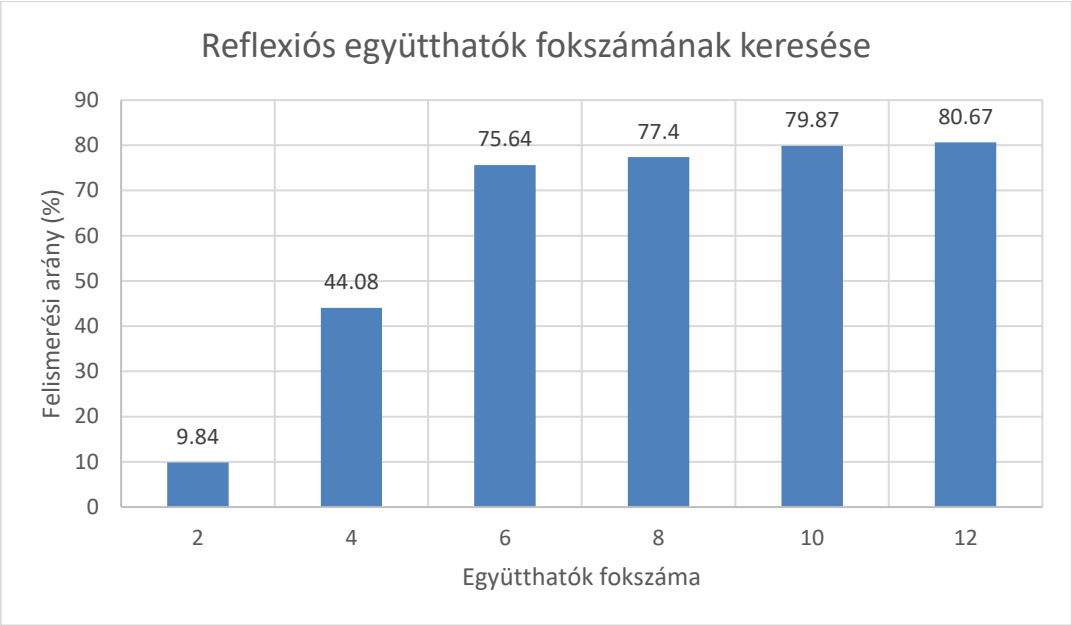
A 4-5. fejezetben látható volt, hogy az LPC által előállított burkoló a fokszám növelésével egyre jobban követi a spektrumot. Azonban az LPC felhasználásának fő oka, hogy figyelmen kívül hagyjuk ezeket a gyors változásokat, és a spektrum jellegét emeljük ki. Ez okból kifolyólag az együtthatóknál főként kisebb számokkal teszteltem a rendszert. A 6-5. ábrán látható a megválasztott intervallum és az egyes értékek közötti lépésköz.



6-5. ábra – LPC fokszámának meghatározása

Az eredmények alapján megállapítható, hogy ha a fokszám értéke 6 alá kerül, akkor a burkoló csak nagyon nagyvonalakban követi a spektrumot, amely nem elégséges információt szolgáltat a rendszer számára. 6 felett pedig túl nagy részletességgel követi azt. Az ideális fokszám LPC esetében a 6.

A reflexiós együtthatók fokszámának meghatározása az előző esettel megegyezően történt, ennek eredménye a 6-6. ábrán látható. Ebben az esetben az ideális fokszám a 12.



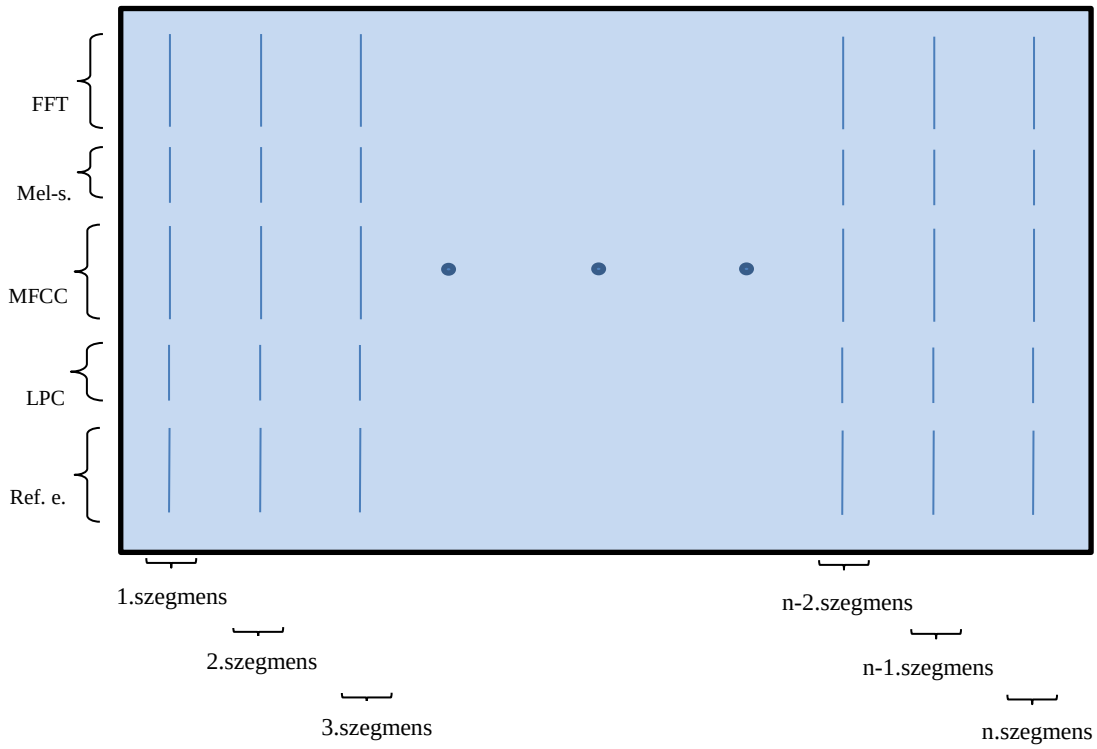
6-6. ábra – Reflexiós együtthatók fokszámának meghatározása

6.1.6. Tanító függvény

A tulajdonságvektorok fúziója során azzal a problémával kerültem szembe, hogy a fejlesztésre használt eszköz nem rendelkezik elegendő memóriával és ezért a tanító függvény nem képes kiszámítani a neurális hálózat megfelelő súlyait és bias értékeit. Korábban említésre is került, hogy a *trainlm* nagyon memória igényes művelet. A probléma elhárítása érdekében újabb tanító függvény találása volt a cél. A választás pedig a szintén matlabban elérhető *trainscg*-re esett, amely lényegesen nem szolgáltatott másabb eredményeket a *trainlm*-től. Gyorsaság szempontjából is kiválóan szerepelt. Részletesebb információk az algoritmus működéséről a [26] találhatók.

6.1.7. Tulajdonságvektorok fúziója

A fúzió során az egyes tulajdonságvektorokat egyesítem, amelynek lényege, hogy a külön-külön jól működő vektorok együttesen több információval szolgáljanak a hálózat számára. A könnyebb érthetőség érdekében a fúzió folyamata a 6-7. ábrán látható. Első lépésként külön-külön, majd a legjobb felismerést eredményezőket egyesítem és úgy vizsgálom a rendszer teljesítményét. A 6-3. táblázat külön-külön felhasznált vektorok eredményeit mutatja.



6-7. ábra – Példa a fúzió folyamatára

Felismerési arány	FFT (%)	CEP (%)	LPC (%)	REF (%)	MFCC (%)	Mel Spektrum (%)
Csecsemő sírás	71.52826	74.04047	44.4444	60.50244243	78.99512	77.0412
Fürdés közben kel-tett mozgások hangja	93.60165	90.60888	9.52	30.85655315	79.66976	88.1321
Autó indítás	100	100	90.48	95.52361201	100	100
Macskanyávogása	56.2123	68.5766	66.1972	49.15560917	56.75513	59.5296
Kutyaugatása	84.80176	90.30837	96.7949	84.36123348	83.9207	79.0749
Férfi/Női beszéd	50.79365	28.18182	81.25	63.49206349	82.31293	85.941
Borotválkozás	95.26316	73.85965	100	98.42105263	94.38596	98.0702
Tusolás	64.45018	67.3768	87.1875	54.57284269	66.60211	77.9212
Porszívózás	73.63465	84.18079	100	75.14124294	94.35028	90.5838
Átlag	76.6984	75.23704	75.1	68.00296133	81.888	84.03

6-3. táblázat – Tulajdonságvektorok eredményei külön-külön

A zölddel jelölt tulajdonságvektorok kerültek felhasználásra, míg a pirossal megjelöltek nem voltak alkalmasak.

A szegmens hosszának megválasztásával két algoritmus kimenetének hossza változik. Az FFT és a Cepstrum által szolgáltatott tulajdonságvektorok a szegmens hosszúságával megegyezőek. Ha a szegmens hossza 2000 és 20000 között lett megválasztva, akkor a neurális hálózat számára ez egy túlságosan nagy bemeneti vektort eredményez. Ennek következménye, hogy a tanítás ideje nagy mértékben megnő. A neurális hálózat kialakítása is egy hosszú folyamat, számomra pedig az elsődleges cél annak megfelelő megtalálása volt viszonylag rövid idő alatt. A 6-3. táblázat rámutat arra is, hogy a Mel spektrum által kiemelt alacsonyabb frekvenciás komponensek a rendszer számára hasznosabbnak bizonyultak az FFT-vel szemben, hiszen a felismerési arány is 10%-al magasabb. A további 4 algoritmus közül, pedig az egyes vektorok egymással való fúziója által kapott rendszer teljesítmények alapján kerültek kiválasztásra. Ezt a 6-4. táblázat mutatja.

Felismerési arány	Mel - MFCC (%)	Mel - MFCC - Ref. (%)	Mel - MFCC - Ref. - LPC(%)
Csecsemő sírás	85.5547	84.1591	83.6706
Fürdés közben keltett mozgások hangja	89.1640	91.9504	91.9504
Autó indítás	100	100	100
Macska nyávogása	67.4909	71.6525	65.1387
Kutya ugatása	80.3964	85.2422	83.9207
Férfi/Női beszéd	80.0453	88.4353	88.4353
Borotválkozás	99.2982	98.0701	98.2456
Tusolás	93.4151	81.9453	95.8037
Porszívózás	90.0188	94.5386	93.2203
Átlag	87.2648	88.4437	88.9317

6-4. táblázat – Tulajdonságvektorok fúziója

A Mel spektrum, az MFCC, a Reflexiós együtthatók és az LPC vektorait egyesítve a rendszer teljesítményét sikerült folyamatosan növelni, így a továbbiakban a fúziót ez a négy algoritmus kimeneti vektorai alkotják.

6.1.8. Rejtett rétegek és hozzájuk tartozó neuronok száma

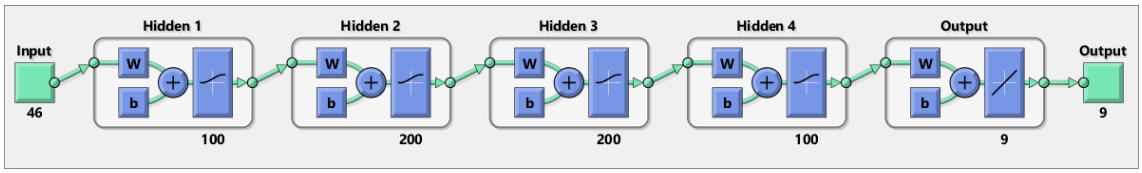
Az eredmények nagyban függnek a neurális hálózat kialakításától, hiszen ez az a pont a rendszerben, ahol a tanulás és osztályozás folyamata lezajlik. A megfelelő rejtett rétegek, illetve neuronok száma szoros kapcsolatban állnak egymással, azok megtalálása is közösen, egymásba ágyazva történt. A megvalósítás előtt bizonyos feltevéseket tettem, ilyen például, hogy a probléma megoldásához elegendő maximum 4 rejtett réteg, illetve az egyes rejtett rétegek neuronszáma nem haladhatja meg a bemeneti vektor hosszának négyszeresét. A megállapításokat egy tesztelési ciklus előzött meg, ahol egy bonyolultabb kialakítású hálózattal kísérleteztem. Az eredmények alapján a túltanulás jelenségét fedeztem fel, amely lényege, hogy a hálózat a mintákban a legapróbb részletekre is ráfókuszál. Ilyen lehet például a hanganyagokban előforduló zaj is. Ennek hatására pedig az eredmények nagy mértékben torzulnak. Így ezekkel a meghatározásokkal leszűkítettem a paraméterek lehetséges értékeinek számát, így belátható hosszúságú iteráció alatt megtalálható volt a megfelelő neurális hálózat kialakítás. A következő 6-5. táblázat a felismerési arány szerint növekvő sorrendben tartalmazza az egyes felhasznált rétegek neuron számát, és az ahhoz tartozó felismerési arányt. Ha a cellába az adott réteghez nem került szám beírva, akkor azt jelenti, hogy a réteg nem került felhasználásra. Vegyük a második értékeket tartalmazó cellát példaként: A neurális hálózat egy rejtett réteget tartalmaz, amelynek 100 neuronja van. A hálózat teljesítménye pedig 88.87%-os. A továbbiakban a legnagyobb felismerési arányt eredményező kialakítást választottam, amely négy rejtett rétegből áll, az első és negyedik réteg száz, míg a második és harmadik réteg kétszáz neuront tartalmaz.

1.rejtett réteg	2.rejtett réteg	3.rejtett réteg	4.rejtett réteg	Felismerési arány (%)
50				22.18323
100				88.87197
100	100			89.52377
100	200	200	200	90.52265
200	100	100		90.54982
100	100	100		90.58731
200	200			90.88003
200	200	100	200	90.90218
100	100	200		90.90943
100	200	100		90.99359
200	200	200	100	91.02664
100	200	200		91.03305
100	200			91.04756
200	100	200	100	91.09447
200	100			91.09462
200	200	100	100	91.12094
100	200	100	200	91.14169
100	100	100	100	91.22916
100	200	100	100	91.26124
100	100	100	200	91.48601
200	100	200	200	91.65341
100	100	200	200	91.73472
200	100	100	200	91.78248
200	200	100		91.79782
200	200	200	200	91.80271
200	100	100	100	92.17632
200	200	200		92.20848
100	200	200	100	92.39381

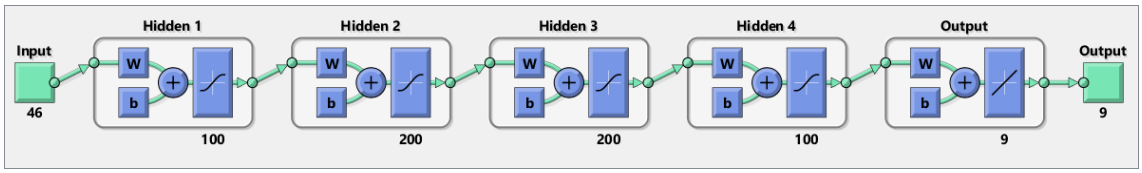
6-5. táblázat – Neurális hálózat kialakítások és azok eredményei

6.1.9. Rejtett rétegek aktivációs függvényei

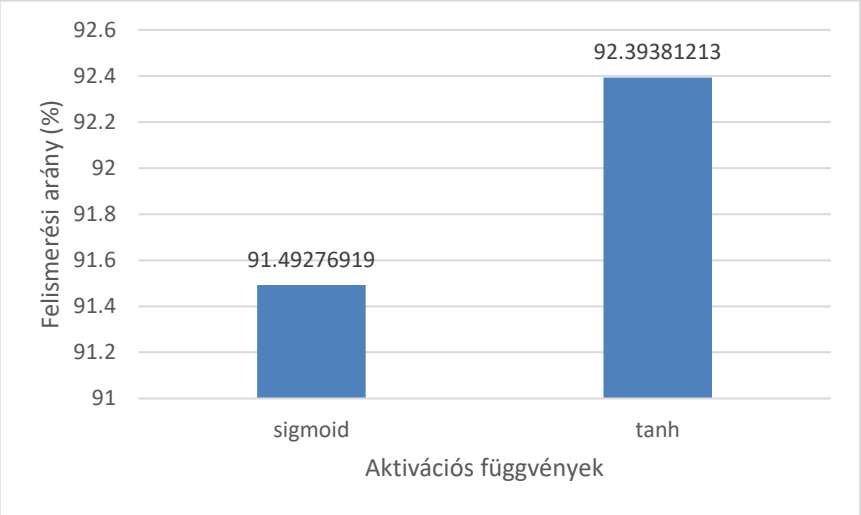
A rejtett rétegek aktivációs függvényeinek változtatásával az egyes neuronok kimeneti értékeinek leképzését szabályozhatjuk. A 5-5. és 5-6. ábrákon a különböző transzferfüggvények, míg a 6-10. ábrán az általuk kapott eredmények láthatók. A kialakított két hálózat a 6-8. és 6-9. ábrákon figyelhetők meg, amelyek felépítésénél figyelembe vettem a 6.1.8-ban kapott eredményeket.



6-8. ábra – Sigmoid aktivációs függvénnyel ellátott neurális hálózat



6-9. ábra - Tanh aktivációs függvénnyel ellátott neurális hálózat

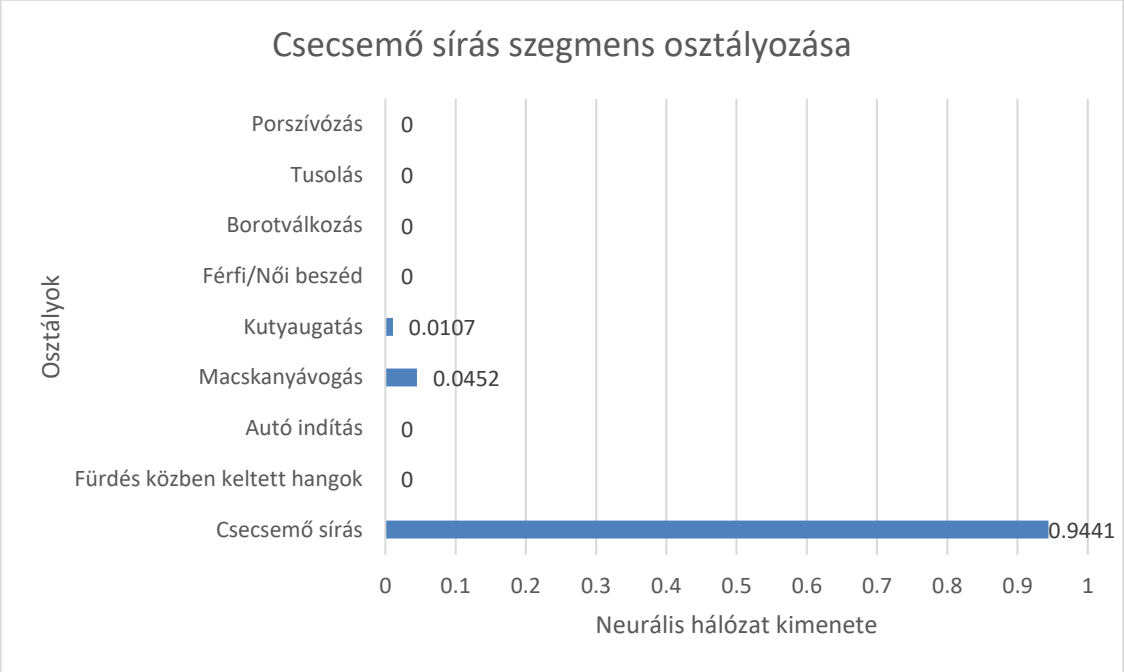


6-10. ábra – Sigmoid és Tanh által kapott felismerési arányok

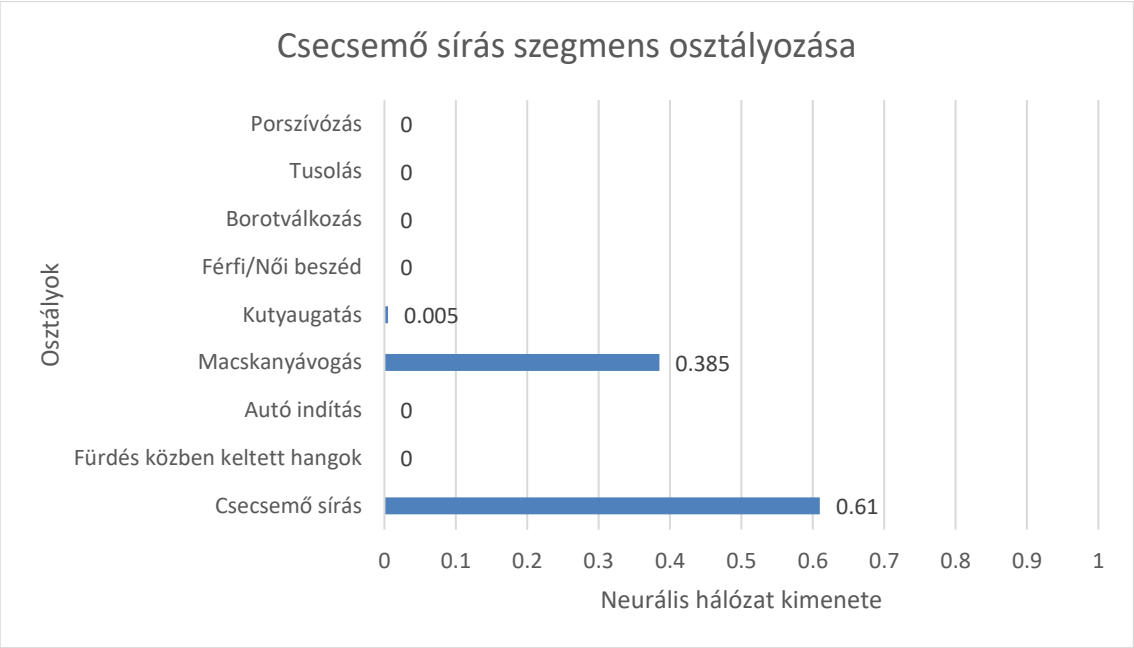
Lényegi különbség nem tapasztalható a két transzfer függvény eredménye között, a tanh használata során 1%-al javult a felismerési arány, így a továbbiakban ezt használtam.

6.1.10. Elfogadási határ

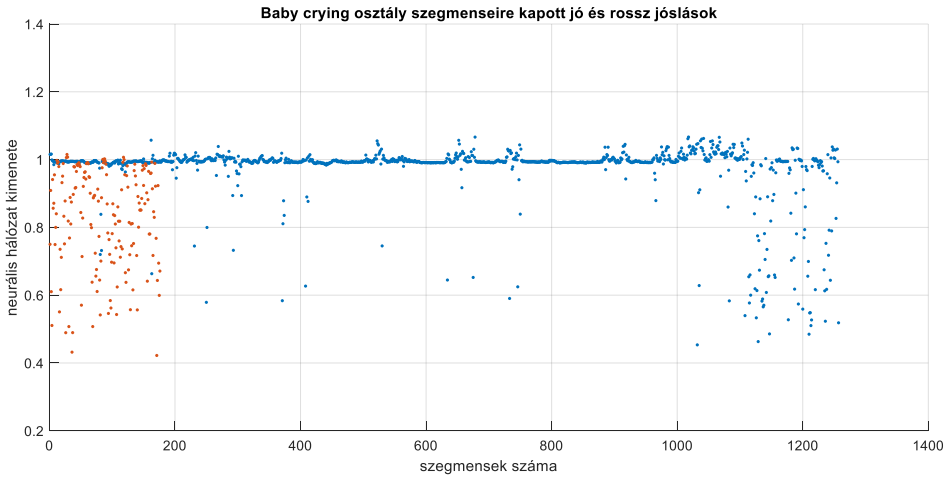
Habár ideális esetben one-hot kódolású kimenetet várunk a neurális hálózattól, valós esetben a kimenet csak közelíti azt. Ez azt jelenti, hogy a kimeneti értékek nem csak 0 és 1, hanem ezektől eltérő értékeket is felvehet. A gyakorlatban a döntést arra az osztályra teszem, ahol a legnagyobb érték található. Egy adott osztályhoz tartozó szegmens biztos osztályozásáról a 6-11. ábrán látható konkrét példa. A tesztadatokra adott válaszokat megfigyelve észrevehetők, hogy a rossz jóslások során a valódi osztályra adott szavazatok is közel vannak a rossz szavazatok értékeihez. Ezt az esetet a 6-12. ábra mutatja. A neurális hálózat által szolgáltatott kimenetet lehetőség van további műveletek felhasználásával javítani. Az egyik lehetőség, hogy azokat a szegmenseket, amelyek nem egyértelműen határozzák meg, hogy mely osztályba tartoznak, tehát van két olyan osztály, amelyre adott jóslás különbsége nem halad meg egy minimum értéket eldobjuk. Ezáltal nem torzítjuk negatív irányba az eredményeket. A másik lehetőség, hogy az összes szegmens által szolgáltatott szavazatot figyelembe vesszük, majd megállapítunk egy olyan elfogadási határt, amely felett a jóslások értékeinek döntő többsége tartozik. Az ez alá eső értékeket nem vesszük figyelembe. Megfigyelhetők a 6-13. ábrától a 6-21. ábráig az egyes osztályokhoz tartozó szegmensekre adott jóslások eredményei. Piros színnel azok a szegmensek láthatók, amelyek prognózisa rossz kimenethez, míg kékkel a jó kimenethez vezettek.



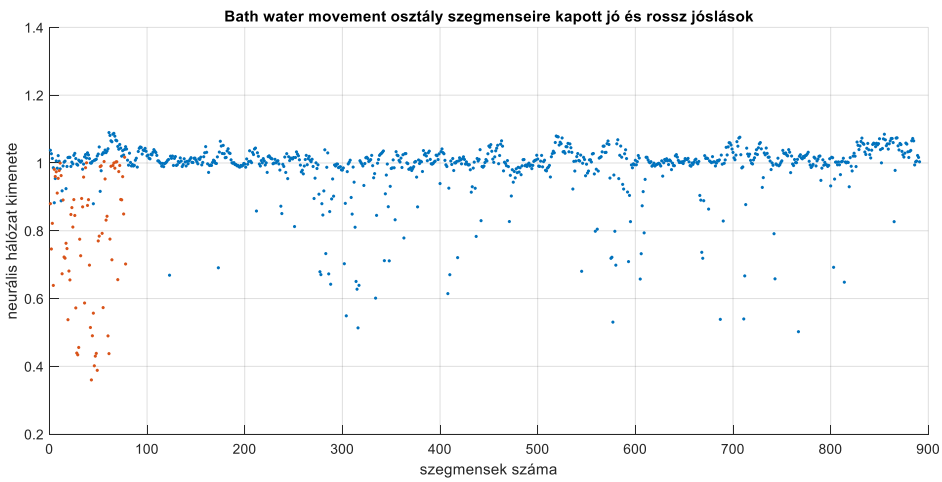
6-11. ábra – Biztos döntés Csecsemő sírás szegmensre



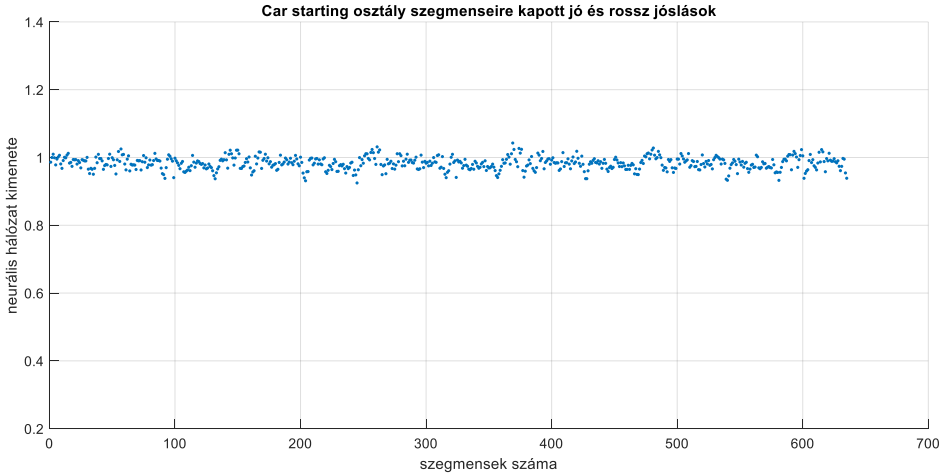
6-12. ábra – Nem biztos döntés Csecsemő sírás szegmensre



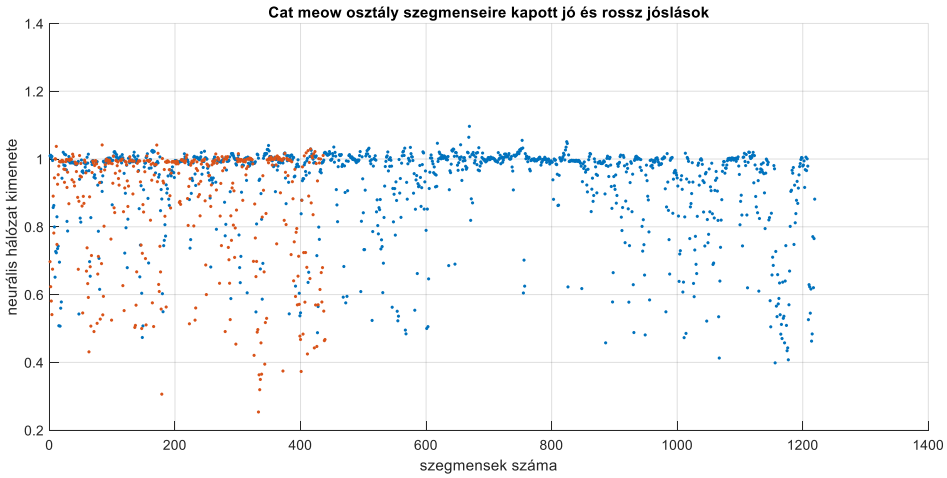
6-13. ábra – Csecsemő sírás szegmenseire kapott jó és rossz jóslások



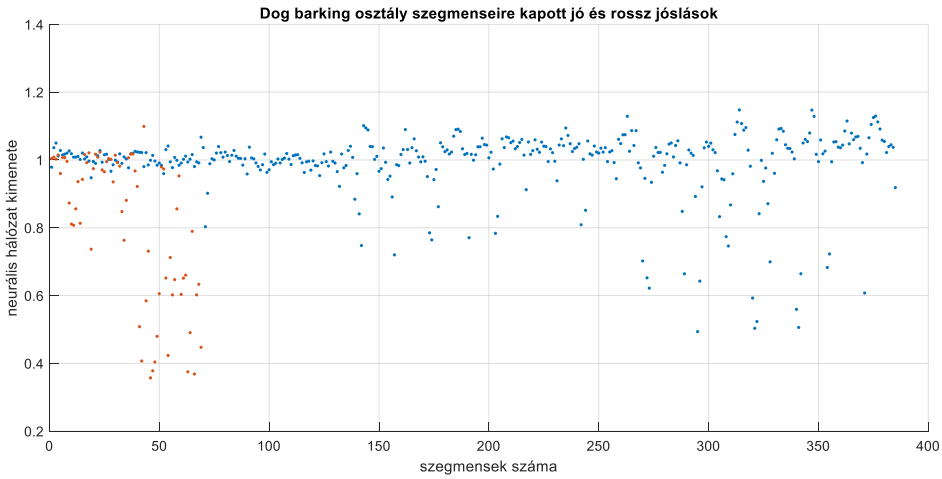
6-14. ábra - Fürdés hangja szegmenseire kapott jó és rossz jóslások



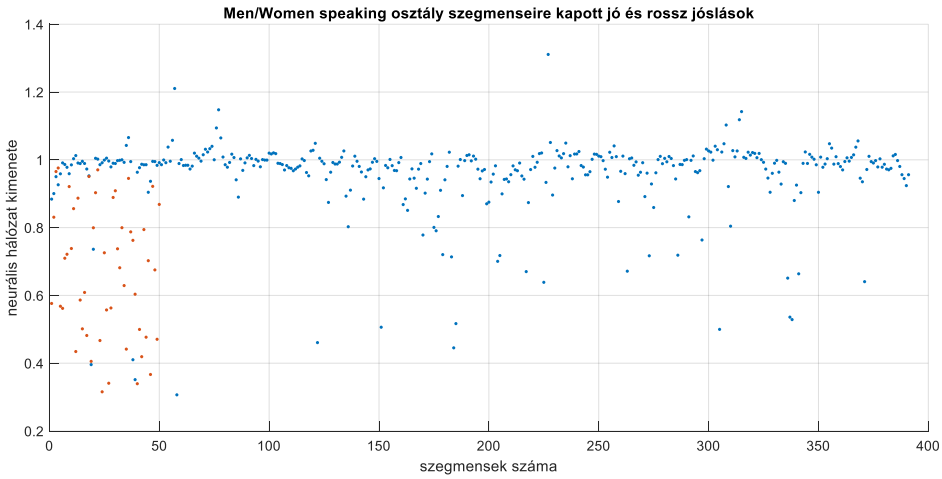
6-15. ábra – Autó indulás szegmenseire kapott jó és rossz jóslások



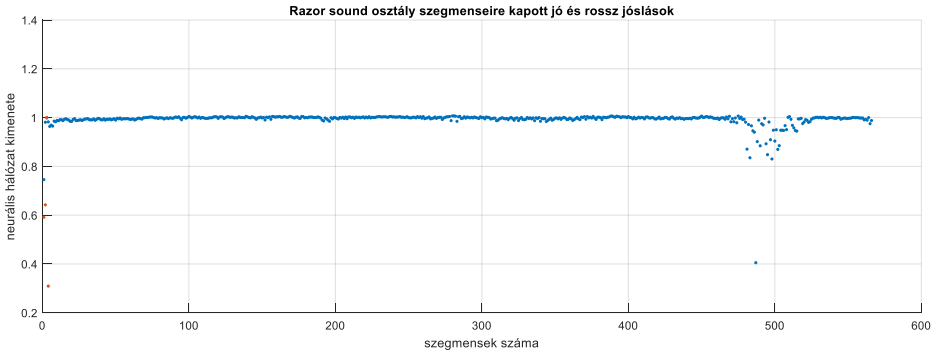
6-16. ábra – Macskanyávogás szegmenseire kapott jó és rossz jóslások



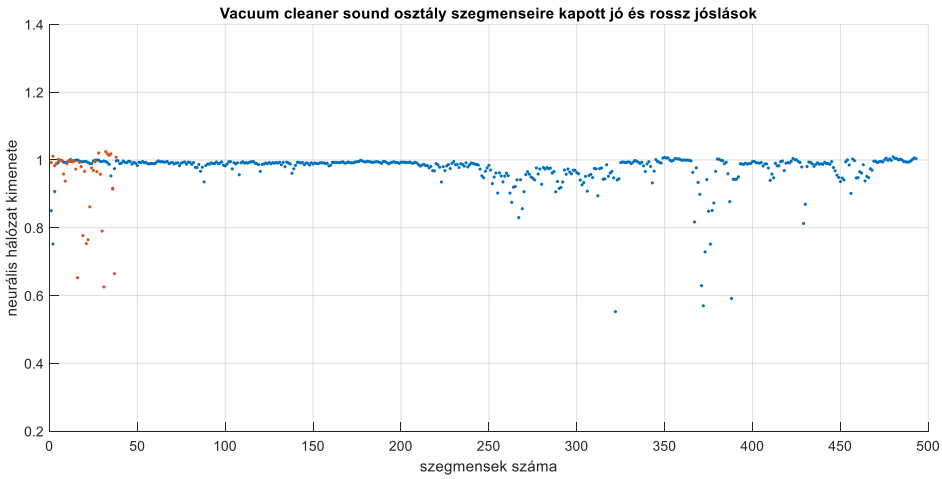
6-17. ábra – Kutyaugatás szegmenseire kapott jó és rossz jóslások



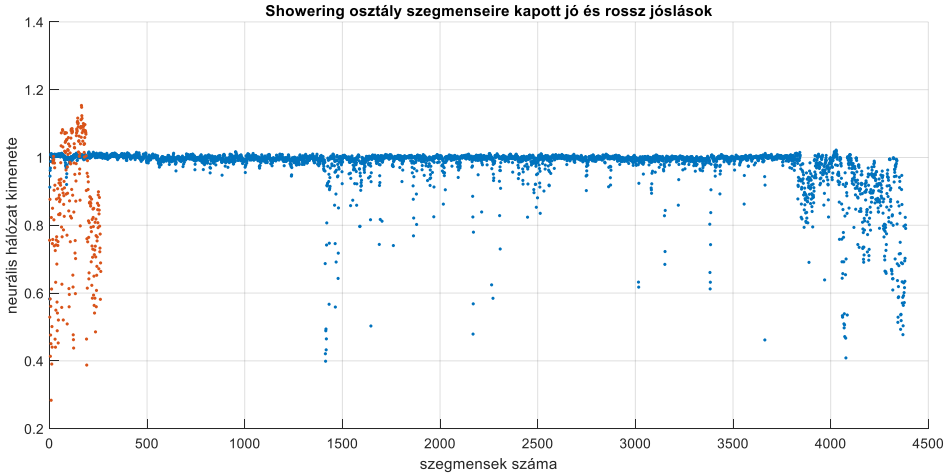
6-18. ábra – Férfi/Női beszéd szegmenseire kapott jó és rossz jóslások



6-19. ábra – Borotválkozás szegmenseire kapott jó és rossz jóslások

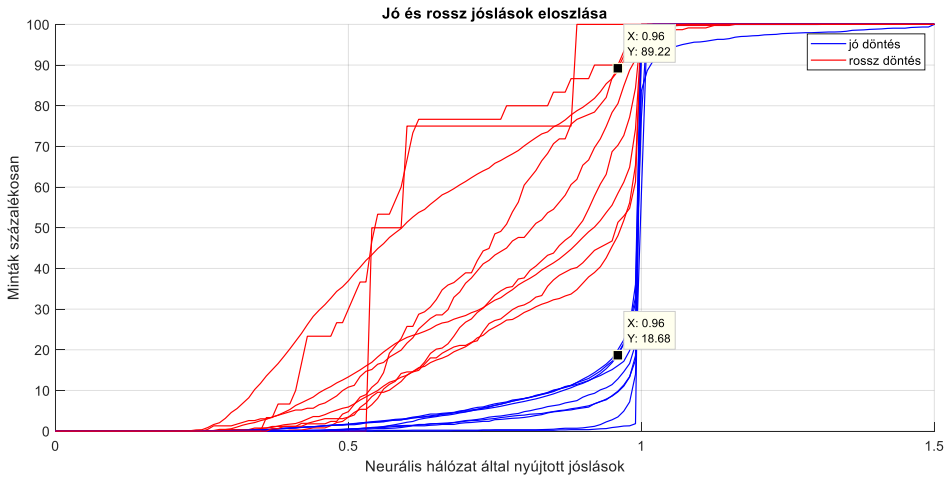


6-20. ábra – Porszívózás szegmenseire kapott jó és rossz jóslások



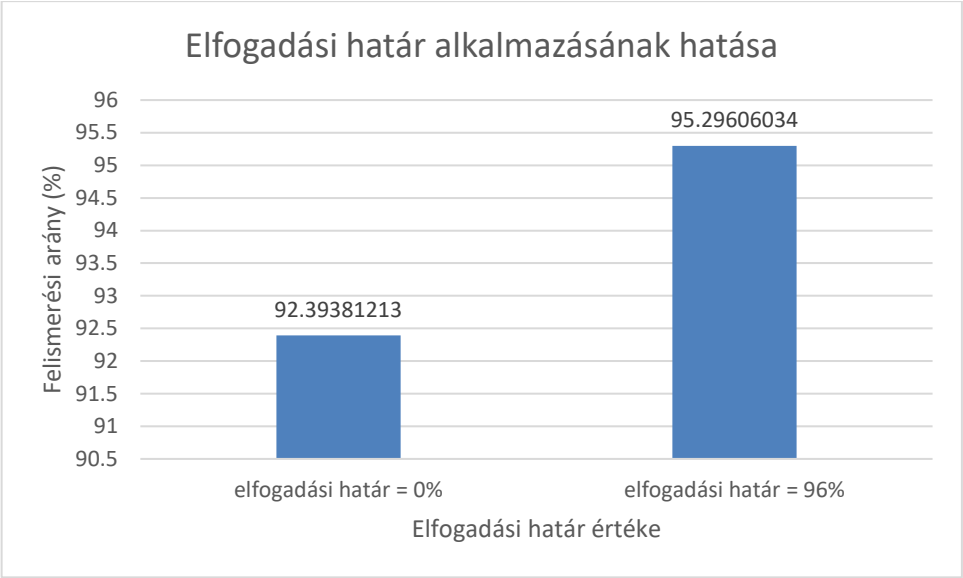
6-21. ábra – Tusolás szegmenseire kapott jó és rossz jóslások

A fentebbi ábrákon észrevehető, hogy a jó válaszokat hordozó szegmensek döntő többsége 1 köré csoportosul. A rosszak pedig elszórtan helyezkednek el. Megválasztható egy olyan határ, amely során a rosszak döntő többsége úgy zárható ki, hogy a jó válaszokat hordozók kisebb része kerüljön eldobásra. Ezt a megállapítást részletesebben mutatja a 6-22. ábra.



6-22. ábra – Jó és rossz jóslások eloszlása

Pirossal itt is a rossz, kékkel pedig a jó prognózis figyelhető meg. Az y tengelyen a mintákat százalékosan, x tengelyen a neurális hálózat kimenetét ábrázoltam. A 6-22. ábrán az egyes kirajzolt kék és piros görbék, az egyes osztályokhoz tartoznak. Azt mutatja meg egy konkrét osztály esetében a görbe adott pontja, hogy a teljes osztályhoz tartozó szegmensek hány százaléka, milyen rendszer kimenetet nyújt. Megfigyelhető, hogy jó jóslások esetében az adott osztályhoz tartozó szegmensek 18.86%-a 0.96-os hálózati kimenetet eredményez, míg rossz jóslások esetében a szegmensek 89.22%-a okoz ugyanakkora kimeneti értéket. Ha az elfogadási határt 0.96-nak választjuk meg, tehát minden olyan szegmenst, amely a hálózat kimenetén 0.96 vagy jobb értéket eredményez megtartunk, az összes többit eldobjuk, akkor a jó szegmensek 18.68%-a vész el, ezzel szemben a helytelen minták 89.22%-a. Az elfogadási határ alkalmazásának eredményét a 6-23. ábra mutatja.



6-23. ábra – Az elfogadási határ alkalmazásának hatása

6.1.11. Paraméterhangolás eredménye

A 6-6. táblázatban összefoglaltam, hogy az osztályozási folyamat paraméterhangolásai és az utófeldolgozás eredőben milyen hatékonyságnövelést eredményezett:

Módszer	Felismerési százalék	Hivatkozás
Alap kialakítása	84.03%	6-3. táblázat
Fúzió	88.93%	6-4. táblázat
NN hangolás	92.39%	6-5. táblázat és 6-10. ábra
Utófeldolgozás	95.29%	6-23. ábra

6-6. táblázat – A paraméterhangolás eredményei

Látható, hogy habár az egyes paraméterhangolási lépések önmagukban viszonylag csekély mértékben javítják a felismerési arányt, összességében viszont több mint 10%-os javulást sikerült elérni különféle technikák segítségével.

6.2. A hálózat eredményeinek értelmezése

A paraméterek meghatározása után létrejött a végső rendszer, amely már képes 90% feletti valószínűséggel megállapítani helyesen a bemeneti mintákról, hogy azok mely osztályhoz rendelhetők. A hálózat működését a végső konstrukcióban kétféle tesztadattal is részletesen elemeztem (lásd 3.1 fejezet). Az egyik tesztadatsor, amellyel az előzetes paraméterhangolást és validációt végeztem, a másik tesztadatsor pedig egy olyan adathalmaz, amelyet mindeddig semmilyen céllal nem használtam. Ez utóbbi teszthalmaz esetén kapott eredmények jellemzik várhatóan legjobban egy valós környezetben használt alkalmazás felismerési arányait.

A paraméterhangoláshoz felhasznált tesztadatok alapján készített confusion mátrix a 6-7. táblázatban található.

%		Felismerendő								
		Csecsemő sírás	Fürdés közben keltett hangok	Autó indítás	Macskanyávogás	Kutyaugatás	Férfi/Női beszéd	Borotválkozás	Tusolás	Porszívózás
Minek ismerte fel	Csecsemő sírás	95.85	0.30	0	14.06	0	0	0	0	0
	Fürdés közben keltett hangok	0.08	96.69	0	0	0	0	0	0	4.15
	Autó indítás	0	0	100	0	0	0	0	2.70	0
	Macskanyávo-gás	4.05	0	0	82.81	8.44	0	0	0.39	0
	Kutyaugatás	0	0	0	0	91.55	0	0	0	0
	Férfi/Női beszéd	0	0	0	1.65	0	100	0	1.83	0
	Borotválkozás	0	0	0	0	0	0	100	0.15	0
	Tusolás	0	0	0	0	0	0	0	94.90	0
	Porszívózás	0	3.00	0	1.47	0	0	0	0	95.84

6-7. táblázat – A rendszerhez tartozó confusion mátrix (1. teszthalmaz)

A hálózat minden osztály esetén 80% feletti teljesítménnyel dolgozott. Egy kiugró eredmény tapasztalható a macskanyávogás mintáinál. A minták 14.05%-nál gondolja azt a rendszer, hogy az a csecsemő sírás csoportjába tartozik. Sok esetben még emberi fül számára is nehéz elkülöníteni ezt a két halmazt, így elképzelhető, hogy a beadott hanganyagok tényleg kellően hasonlítanak a csecsemő síráshoz. A többi osztály esetében azonban kitűnő eredményekkel szolgált a rendszer.

Vizsgáljuk meg most, milyen a hálózat működése általános esetben. Itt a tanított hanganyagtól eltérő típusú minták is szerepet kaptak. A 6-8. táblázat a legelőször kialakított neurális hálózat eredményeit mutatja, amely egy 100 neuront tartalmazó rétegből állt.

%		Felismerendő								
		Csecsemő sírás	Fürdés közben keltett hangok	Autó indítás	Macskanyávogás	Kutyaugatás	Férfi/Női beszéd	Borotválkozás	Tusolás	Porszívózás
Minek ismerte fel	Csecsemő sírás	79	13.17	0	12.12	5.69	0.44	10.68	0	0.35
	Fürdés közben keltett hangok	3.17	43.96	56.07	1.51	3.25	4.65	28.65	1.19	67.11
	Autó indítás	0	0.61	7.08	0	0	0.44	0	1.38	0.11
	Macskanyávo-gás	14.87	34.97	30.23	86.36	21.95	8.59	48.25	0	2.35
	Kutyaugatás	0.92	0.24	0	0	60.97	0.08	0	1.19	0
	Férfi/Női beszéd	1.47	0.12	6.21	0	0.81	81.73	0	2.77	0.03
	Borotválkozás	0	2.58	0	0	0	0.08	12.12	0	0
	Tusolás	0	3.69	0.03	0	0	0.62	0.27	93.45	0.99
	Porszívózás	0.54	0.61	0.34	0	7.31	3.31	0	0	29.02

6-8. táblázat – Egyszerű neurális hálózathoz tartozó confusion mátrix (2. teszthalmaz)

Észrevehető, hogy a hálózat sok esetben hibázik. Az Autó indítás osztály esetében 56.07%-osan állítja, hogy az a Fürdés közben keltett hangok osztályába tartozik. Illetve a Borotválkozás és Porszívózás mintáinál is rossz válaszokat kaptunk. Azonban mi a helyzet a kialakított neurális hálózattal? Mennyire sikerült általánosítani a rendszert a paraméterek hangolásával? Erre ad választ a 6-9. táblázat.

%		Felismerendő								
		Csecsemő sírás	Fürdés közben keltett hangok	Autó indítás	Macskanyávogás	Kutyaugatás	Férfi/Női beszéd	Borotválkozás	Tusolás	Porszívózás
Minek ismerte fel	Csecsemő sírás	89.85	19.21	0.06	6.06	0	1.79	17.68	0	0.09
	Fürdés közben keltett hangok	2.09	48.17	37.31	0.75	1.62	0.08	13.25	0	54.05
	Autó indítás	0	0.61	47.56	0	0	0.44	0	1.19	0.15
	Macskanyávo-gás	6.97	24.97	6.31	92.42	38.21	4.29	31.76	0.79	0.03
	Kutyaugatás	0.15	0	0	0	57.72	0.17	0	0	0
	Férfi/Női beszéd	0.92	0	8.65	0	2.434	91.58	0.17	3.37	0.67
	Borotválkozás	0	0.61	0	0	0	0	36.48	0	0
	Tusolás	0	6.03	0	0.75	0	0.35	0.632	94.64	1.35
	Porszívózás	0	0.36	0.06	0	0	1.25	0	0	43.62

6-9. táblázat – A kialakított rendszerhez tartozó confusion mátrix (2. teszthalmaz)

Látható, hogy többségében az átlóban helyezkednek el a legnagyobb értékek, amelyet zölddel jelöltem. Található egymáshoz nagyon közeli érték is. Ilyen például a Borotválkozás osztály szegmenseire adott válasz is, amely 36,48%-ban Borotválkozás, míg 31.75%-ban Macskanyávogás halmazába lett sorolva. A 6-8. táblázatban a Borotválkozás mintáinak felismerési aránya 12.12% volt. Az Autó indítás esetében is rossz választ kaptunk, azonban a paraméterek hangolása után itt is sikerült nagyobb százalékban helyes értéket előállítani.

Összességében elmondható, hogy a rendszer hangolása több esetben is javulást eredményezett. Egy valós alkalmazásban történő felhasználáshoz a rendszer felismerőképességét még célszerű javítani, de a fenti tesztek biztató eredményekkel szolgálnak.

7. Konklúzió, további fejlesztési lehetőségek

A teljes folyamat a rendszert alkotó elemek megismerésével kezdődött. A működéshez nélkülözhetetlen adatbázis kialakításával, amelyet felhasználtunk a tanítására, illetve tesztelésére. Az algoritmusok implementálása után a rendszerben szereplő paraméterek meghatározása következett, amelyet megelőzött egy neurális hálózat előre definiálása. A rendszer számára legtöbb információt hordozó tulajdonságvektorok kialakítása után, az osztályozó hangolása következett. A végső hálózat által szolgáltatott válaszokból némi utófeldolgozással még jobb eredményeket állítottam elő. Összességében sikerült egy jól működő rendszert megalkotni, amellyel képesek vagyunk a rendszer bemenetére adott hanganyagról átlagosan 95%-os pontossággal megállapítani, hogy az a definiált osztályok közül melyikbe sorolható. Általános helyzetekben ez az érték 66.89%, amelyet nagymértékben javítani az adatbázis növelésével lehetséges.

A témám során elsajátíthattam különböző jelfeldolgozási technikákat, megismerhettem mik a manapság leghasználatosabb algoritmusok ezen a területen. Betekintést nyerhettem a mesterséges intelligencia világába, majd az így szerzett tudás felhasználásával létrehoztam saját neurális hálózatomat, és megvalósítottam a célként kitűzött alkalmazást.

A hasonló rendszerekkel szemben támasztott követelmények között általában szerepel a valós idejű működés. Ennek implementálása megtörtént, azonban az eredmények azt mutatták, hogy ha a tanító adathalmaz felvételének módja eltér a teszt adathalmaz felvételének módjától, akkor nem tökéletes eredményeket kapunk. Ahhoz, hogy az alkalmazás egyszerű laptop által használt mikrofonok segítségével működhessen, létre kellene hozni egy adatbázist, amelyet szintén hasonló mikrofonnal rögzítenek. Az adatbázis mérete is fontos szerepet tölt be a rendszer általános működésében, ahogy az korábban látható volt. Annak növelése elengedhetetlen a jobb eredmények elérése érdekében. Ezek a feladatok mind a továbbfejlesztési lehetőségek körébe sorolhatók.

Köszönetnyilvánítás

Szeretném ezúton is megköszönni konzulensemnek, dr. Orosz Györgynek a kitartó és támogató munkáját, hogy tudásával és hasznos tanácsaival segítette szakdolgozatom létrejöttét.

Az alkalmat megragadva szeretném kifejezni hálámat szüleimnek az önzetlen támogatásukért, és amiért megteremtették számomra a tanulás lehetőségét.

Irodalomjegyzék

[1] Audacity. 2018. audacityteam.org. URL: https://www.audacityteam.org/. [utolsó hozzáférés: 2018 december 3].

[2] Zapsplat. 2018. zapsplat. URL: https://www.zapsplat.com/. [utolsó hozzáférés: 2018 december 3].

[3] F. Harris, "On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform," in Proc. IEEE, vol. 66, no. 1, pp. 51-83, Jan. 1978William Strunk Jr., E. B. White, *The Elements of Style*, Fourth Edition, Longman, 4th edition, 1999.

[4] Mathworks. 1994-2018. *Deep Learning Toolbox.* URL: https://www.mathworks.com/products/deep-learning.html. [utolsó hozzáférés: 2018 december 3].

[5] Mathworks. 1994-2018. *Parallel Computing Toolbox*. URL: https://www.mathworks.com/products/parallel-computing.html. [utolsó hozzáférés: 2018 december 3].

[6] Dr. Huba Antal. 2014. *Mechatronika, Optika és Gépészeti Informatika tanszék – Méréselmélet 6. fejezet*. URL: http://www.mogi.bme.hu/TAMOP/mereselmelet/ch06.html. [utolsó hozzáférés: 2018 december 3].

[7] Jason Brownlee. 2016. *What is a Confusion Matrix in Machine Learning.* URL: https://machinelearningmastery.com/confusion-matrix-machine-learning/. [utolsó hozzáférés: 2018 december 3].

[8] Kovács György. 2011. *A jelfeldolgozás matematikai alapjai.* 5. fejezet. URL: https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0103_02_jelfeldolgozas_matematikai_alapjai/ch05.html. [utolsó hozzáférés: 2018 december 3].

[9] James R. Andrews, M. Gerald Arthur, *„Spectrum Amplitude – Definition, Generation and Measurement",* pp. 4-5, publisher: U.S. Government Printing Office, Washington, 1977.

[10] Dr. Huba Antal. 2014. *Mechatronika, Optika és Gépészeti Informatika tanszék – Méréselmélet 11. fejezet*. URL: http://www.mogi.bme.hu/TAMOP/mereselme_let/ch011.html. [utolsó hozzáférés: 2018 december 3].

[11] Wikipedia. 2018. *Fourier-transzformáció.* URL: https://hu.wikipedia.org/wiki/Fourier-transzform%C3%A1ci%C3%B3. [utolsó hozzáférés: 2018 december 3].

[12] Abhilisha Sukhwal, Mahendra Kumar. 2015. *Comparative stury of different classifiers based speaker recognition system using modified MFCC for noisy environment*. pp. 977, URL: https://www.researchgate.net/publication/304292833_Comparative_study_of_different_classifiers_based_speaker_recognition_system_using_modified_MFCC_for_noisy_environment. [utolsó hozzáférés: 2018 december 3].

[13] Haytham Fayek. 2016. *Speech Processing for Machine Learning*. URL: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. [utolsó hozzáférés: 2018 december 3].

[14] Purnima Pandit, Shardav Bhatt, „*Automatic Speech Recognition of Gujariti digits using Radial Basis Function Network",* pp. 4-6, publisher: A D Publication, India, 2016.

[15] Wikipedia. 2018. *Cepstrum.* URL: https://en.wikipedia.org/wiki/Cepstrum. [utolsó hozzáférés: 2018 december 3].

[16] M. A. Pathak, *„Privacy-Preserving Machine Learning for Speech Processing"*, pp. 7, publisher: Springer Science+Business Media, New York, 2013.

[17] John Makhoul, *„Linear Prediction: A Tutorial Review"*, Vol. 63, pp. 1-20, No. 4, publisher: Proceedings of the IEEE, 1975.

[18] Balogh Bertalan. 2011. *Keresztkorreláció vizsgálata statisztikai teszttel*. pp. 9-11. URL: http://phys.chem.elte.hu/test/szakdolik/2011/2011_BaloghBertalan_KemiaBSc.pdf. [utolsó hozzáférés: 2018 december 3].

[19] Vermes Mátyás, *„Akusztikus impedancia becslése szeizmikus csatornák spektrumának extrapolációjával".* Magyar Geofizika XXVII. Évf. 3-4. szám, Budapest, 1986.

[20] Mathworks. 1994-2018. *Levinson.* URL: https://www.mathworks.com/help/signal/ref/levinson.html. [utolsó hozzáférés: 2018 december 3].

[21] Tavish Srivastava. 2018. *Introduction to k-Nearest Neighbors: Simplidied (with implementation in Python).* URL: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/. [utolsó hozzáférés: 2018 december 3].

[22] Altrichter Márta, Horváth Gábor, Pataki Béla, Strausz György, Takács Gábor, Valyon József, „*Neurális hálózatok"*. 1. fejezet. publisher: Panem Könyvkiadó Kft., Budapest, 2006.

[23] Wikipedia. 2018. *Perceptron*. URL: https://en.wikipedia.org/wiki/Perceptron. [utolsó hozzáférés: 2018 december 3].

[24] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, *„Bevezetés az adatbányászatba”*. URL: https://www.tankonyvtar.hu/hu/tartalom/tamop425/0046_adatbanyaszat/ch05s04.html. [utolsó hozzáférés: 2018 december 3].

[25] Michael A. Nielsen, *„Neural Networks and Deep Learning”,* 1. fejezet. publisher: Determination Press, 2015.

[26] Martin Fodslette Moller, „Neural Networks”, Vol. 6. pp. 525-533, publisher: Pergamon Press Ltd., USA, 1993