Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Gráfkonvolúciós hálók tervezése és alkalmazása molekulaszerkezetek vizsgálatában

TDK dolgozat

Készítette:

Pelle Mátyás

Konzulens:

Dr. Szegletes Luca

2020

Tartalomjegyzék

Kivonat i

Abstract ii

1. Bevezetés 1
1.1. Mesterséges intelligencia és a neurális hálózat . . . 1
1.2. A feladat fontossága . . . 2
1.3. A dolgozat felépítése . . . 2

2. Szakirodalom áttekintése, kapcsolódó kutatások, adathalmazok 3
2.1. Molekulák . . . 3
2.1.1. SMILES stringek [1] . . . 4
2.1.2. Fingerprint [2] . . . 4
2.2. Adathalamzok . . . 5
2.2.1. TTK molekulák . . . 5
2.2.2. QM7b [3] [4] [5] . . . 6
2.3. MoleculeNet [6] . . . 6

3. Módszerek 9
3.1. GCN . . . 9
3.1.1. GCN mintapéldák . . . 9
3.1.2. A GCN elmélete . . . 9
3.1.2.1. Feltételek [7] . . . 10
3.1.2.2. Nehézség a képfeldolgozással szemben . . . 11
3.1.3. Matematikai háttér [8] [9] . . . 11
3.2. 3DGCN [10] . . . 13
3.2.1. A modell leírása . . . 13
3.2.2. A **3DGCN** moduljai . . . 14
3.3. A 3DGCN kibővítései . . . 15
3.3.1. A tanítási, validációs és teszthalmaz nagyságának beállítása . . . 15
3.3.2. Augmentálás elvégzése . . . 16
3.3.3. Konkrét tanítási, validációs és teszt halmaz megadása . . . 16
3.4. 3DGCN és autoenkóder . . . 17
3.4.1. Autoenkóder [11] . . . 17
3.4.2. 3DGCN AE . . . 18

4. Vizsgálatok, számítások menete 19
4.1. Redukciós potenciál . . . 19
4.1.1. Számolási protokoll . . . 20
4.2. Fájl konverzió, xyz, sdf . . . 21
4.2.1. xyz . . . 21

4.2.2. sdf . . . 22
4.3. Hiperparaméterek optimalizálása . . . 23
4.3.1. Hiperparaméterek optimalizálás folyamata . . . 23

5. Eredmények **26**
5.1. 3DGCN . . . 26
5.1.1. A teljes TTK adathalmazon . . . 26
5.1.2. Nem a teljes TTK adathalmazon . . . 27
5.2. 3DGCN AE . . . 28
5.2.1. A teljes TTK adathalmazon . . . 28
5.2.2. Nem a teljes TTK adathalmazon . . . 29
5.3. A tanítási, validációs és a teszthalmaz nagyságának következményei . . . 30
5.3.1. Tanítási 60%, validációs és tesztelési halmaz 20% . . . 30
5.3.1.1. 3DGCN . . . 30
5.3.1.2. 3DGCN AE . . . 31
5.3.2. Tanítási 40%, validációs és tesztelési halmaz 30% . . . 31
5.3.2.1. 3DGCN . . . 31
5.3.2.2. 3DGCN AE . . . 32
5.3.3. Tanítási 20%, validációs és tesztelési halmaz 40% . . . 32
5.3.3.1. 3DGCN . . . 32
5.3.3.2. 3DGCN AE . . . 33
5.4. QM7b . . . 33
5.4.1. 3DGCN . . . 33
5.4.2. 3DGCN AE . . . 34
5.5. FreeSolv . . . 34
5.5.1. 3DGCN . . . 34
5.5.2. 3DGCN AE . . . 35
5.6. ESOL . . . 35
5.6.1. 3DGCN . . . 35
5.6.2. 3DGCN AE . . . 35

6. Összefoglalás **37**
6.1. Továbbfejlesztési lehetőségek . . . 37

Köszönetnyilvánítás **38**

Irodalomjegyzék **39**

Kivonat

Az utóbbi években a gépi tanulás robbanásszerű fejlődésével egyre több tudományágban jelent meg a deep learning (mélytanulás) és használják azóta az eszköztárát különböző problémák megoldására kezdve a gépi látástól az önvezető autókon át az ajánlórendszerekig.

Azonban a mélytanulás alkalmazása gráfokon nem triviális a gráfok egyedi szerkezete miatt. A gráfoknak egy alkalmazási területe a szerkezeti kémia. Itt gráfokkal modelleznek molekulaszerkezeteket és vizsgálják a gráf és a molekula tulajdonságai közötti összefüggéseket. A gráf pontok (például atomok) és élek (például az atomok közötti kötések) halmaza. Megfigyelték, hogy a molekulák fizikai és kémiai tulajdonságai és a kémiai képletük topológiai szerkezete között összefüggés van (pl. szénhidrogén vegyületek szerkezeti képlete és forráspontja között). Így, ha ismerjük egy molekula topológiai szerkezetét, következtethetünk annak tulajdonságaira.

Kutatásom célja olyan gráfkonvolúciós hálózatok vizsgálata, melyeknek alkalmazási területe a szerkezeti kémia. Megvizsgálom a szerkezeti kémiában ismert különböző molekula reprezentációkat, bemutatom a gráf alapú modellek előnyeit és hátrányait, különös tekintettel a kétdimenziós és a háromdimenziós megvalósításokra. A háromdimenziós gráfkonvolúciós háló több információt tartalmaz, mint a kétdimenziós reprezentáció, hiszen rendelkezik az atomok térbeli koordinátáival. Dolgozatom a háromdimenziós megoldásra fókuszál. A gráfkonvolúciós hálózatok segítségével megvizsgálom a gráf alapú háromdimenziós molekula szerkezeteket. Ezt a modellt egészítem ki egy saját magam által tervezett és megvalósított autoencoderrel, melynek célja a hálózat optimalizálása. Az algoritmus célja háromdimenziós molekulaszerkezetekből molekulatulajdonságok becslése, ugyanis ezek általában időigényes kvantumkémiai számolást igényelnek. Megmutatom, hogy a neurális hálózat segítségével gyorsabban el lehet végezni ezeket a számolásokat a pontosság megtartásával. Valós kémiai adatokon tesztelem az algoritmust, elemzem az így kapott eredményeket.

Abstract

During recent years, with the explosive development of machine learning, deep learning has appeared in more and more branches of science and its toolkit has since been used to solve a variety of problems ranging from machine vision through self-driving cars to recommendation systems.

However, the application of deep learning on graphs is not trivial due to the unique structure of the graphs. One area of application for graphs is structural chemistry. In chemistry they model molecular structures with graphs and study the correlations between the attributes of the graph and the molecule, where the nodes of the graph represent atoms and the edges of the graphs represent atomic bonds. It has been observed that there is a correlation between the physical and chemical properties of molecules and the topological structure of their chemical formula (e.g. between the structural formula and boiling point of hydrocarbon compounds). This way if we know the topological structure of a molecule, we can deduce its properties.

The goal of my research is to study graph convolutional networks, the field of application of which is structural chemistry. I examine the different molecular representations known in structural chemistry and present the advantages and disadvantages of graph-based models, with special regard to two-dimensional and three-dimensional implementations. The three-dimensional graph convolution network contains more information than the two-dimensional representation because it has the spatial coordinates of the atoms. My dissertation focuses on the three-dimensional solution. I investigate graph-based three-dimensional molecular structures with the use of graph convolutional networks. I supplement this model with an autoencoder, which is designed and implemented by myself, to optimize the network. The aim of the algorithm is to estimate molecular properties from three-dimensional molecular structures, as they usually require time-consuming quantum chemical calculations. I will show that the neural network can be used to perform these calculations faster while maintaining accuracy. I test the algorithm on real chemical data and analyse the results obtained in this way.

1. fejezet

Bevezetés

A technológia minden területen folyamatosan fejlődik, melynek következménye a felgyorsuló világ. Ebben a gyorsuló világban nagyon fontos, hogy egy adott tevékenység mennyi ideig tart. Ha valami tovább tart a megszokottnál, elvártnál annak komoly következményei, időveszteségei lehetnek. Mivel a gyorsaság, idő az egyik legfontosabb paraméter, ezért egyre gyakrabban az emberi tényezőt próbáljuk technológiával helyettesíteni. Különböző technológiai megvalósítások, módszerek léteznek. Manapság az egyik legelterjedtebb megoldás a mesterséges intelligencia segítségül hívása, mivel az emberi képességek, kapacitások korlátosak. Természetesen a technológiai erőforrások sem bővíthetők a végtelenségig, vannak azért korlátai.

1.1. Mesterséges intelligencia és a neurális hálózat

Manapság már szinte majdnem mindenre rámondják hogy mesterséges intelligencia[1]. Az MI az egy olyan intelligencia amit egy gép, program mesterségesen hoz létre. Az MI-t több irányból meg lehet közelíteni, 4 lehetséges megközelítés [12] a következők:

- Emberi módon gondolkodó MI: kognitív modellezés
- Emberi módon cselekvő MI: Turing teszt, chat botok
- Racionálisan gondolkodó MI: logika, következtetés
- Racionálisan cselekvő MI: a „megfelelő” dolgot teszi a feladat megoldásához

Az MI az egy nagyon tág fogalom, nagyon sok minden beletartozik. A különböző típusú megközelítéseknek vannak előnyei is hátránya, nem lehet mindegyik fajta megközelítést használni minden problémára, valamikor a kognitív megközelítés kell, máskor pedig a logikai következtetés.

Neurális hálózatokat használnak a mély tanulásnál[2], ami az MI részhalmaza. Neurális hálózatnak nevezzük azt a hardver vagy szoftver megvalósítású párhuzamos, elosztott működésre képes információfeldolgozó eszközt, amely [13]:

- Azonos, vagy hasonló típusú – általában nagyszámú – lokális feldolgozást végző műveleti elem, neuron (processing element, neuron) többnyire rendezett topológiájú, nagymértékben összekapcsolt rendszeréből áll.
- Rendelkezik tanulási algoritmussal (learning algorithm), mely általában minta alapján való tanulást jelent, és amely az információfeldolgozás módját határozza meg.

[1] MI vagy AI, ami az angol Artificial Intelligence-nek a rövidítése
[2] DL - Deep Learning

- Rendelkezik a megtanult információ felhasználását lehetővé tevő információ előhívási, vagy röviden előhívási algoritmussal (recall algorithm).

A neurális hálózatok működését két nagy fázisra lehet felosztani. Tanulási és előhívási fázis, melyek időben szétválnak egymástól, a tanulási fázis tipikusan hosszadalmas folyamat szokott lenni, ezzel szemben az előhívási fázis általában gyors. Az első fázis a tanulási fázis, itt az általunk létrehozott hálózatnak a beadott adathalmaz segítségével mintákat kell megtanulnia, rejtett információkat kell felismernie. Miután a hálózat betanult, kapunk egy rendszert, ami képes felismerni az adott mintákat, ez az előhívási fázis.

1.2. A feladat fontossága

A dolgozatom elkészítése során az volt a feladatom, hogy gráfkonvolúciós hálókat vizsgáljak, tervezzek molekulaszerkezeteken. Redoxpotenciállal jellemezzük azt, ha egy molekula elektront vesz fel, vagy elektront ad le. A redoxpotenciál pontos, magas szintű kvantumkémiai számolása nagy számú molekulákra nagyon időigényes, kivitelezése adott esetben lehetetlenné is válhat. A célom az volt, hogy megkönnyítsem a kémikusok munkáját azzal, hogy mesterséges intelligencia segítségével az adott molekulák redoxpotenciál értékét becsüljem, regressziót végezzek.

1.3. A dolgozat felépítése

A dolgozatom bevezetése után ismertetem a szakirodalmi áttekintést, kapcsolódó kutatásokat. Mindig fontos utánanézni az adott témakörnek, beszerezni a megfelelő információkat, kutatást végezni. Jelen témakörben ezek a molekulák ismertetése, reprezentálása, különböző módszerekkel, adathalmazok, hasonló megvalósítások.

A következő fejezet a legterjedelmesebb része a dolgozatomnak, ugyanis itt ismertetem a módszereket, megvalósításokat. Bemutatom a gráfkonvolúciókat kétdimenziós és háromdimenziós környezetben. Meglévő módszerekkel és kibővítésekkel, autoenkóderrel kombinálva.

Az ezt követő fejezet egy szintén terjedelmes és fontos részt foglal magába. Itt szeretném ismertetni a gyakorlati számítási protokollt, pipelinet és nem utolsó sorban a hiperparaméter optimalizálását. Mindig nagyon fontos elvégezni az optimalizálást, mert lehet hogy egy jó algoritmus nem megfelelő paraméterekkel rossz eredményeket add.

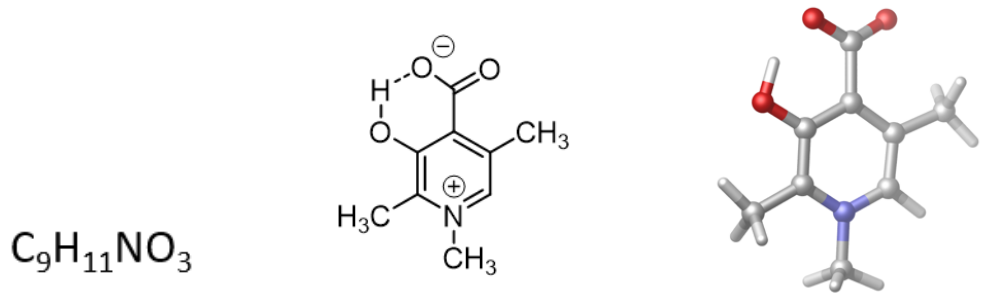
A dolgozatom végén pedig szeretném ismertetni a kapott eredményeket, összehasonlítást végezni, levonni a következményeket. Egy adott témakört egyre mélyebben meglehet ismerni, így ezt is, mindig kínálkozik továbbfejleszti lehetőség.

2. fejezet

Szakirodalom áttekintése, kapcsolódó kutatások, adathalmazok

2.1. Molekulák

Egy molekulának többféle reprezentációja létezik, megkülönböztethetünk grafikai, tartalmi reprezentációkat. Ezek az egyes reprezentációk nem egyforma mennyiségű információt hordoznak az adott rendszerről. A legegyszerűbb megjelenítése az az, amikor a molekulát csak a molekula képlet vagy vegyjel segítségével ábrázolunk. Ha már szerkezeti képlet segítségével ábrázoljuk, akkor már több információ áll a rendelkezésünkre. Itt megkülönböztethetjük, hogy kétdimenziós vagy háromdimenziós szerkezetről van-e szó. Ha rendelkezünk az atomok térbeli elhelyezkedésével, akkor több információ áll a rendelkezésünkre. A három dimenzió több információt hordoz számunkra, mint a két dimenzió. A SMILES stringeket vagy a fingerprint-eket használnak manapság a molekulák digitalizálásában.



(a) Molekulaképlet (b) Szerkezeti képlet (c) Térbeli szerkezet

2.1. ábra. Molekulák reprezentációja

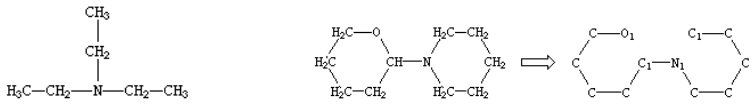
Különböző célokra, különböző reprezentáció alkalmas, mindig a célnak megfelelőt kell választani. A kvantumkémiai számolásokban fontos az atomok térbeli elrendeződésének az ismerete, ezért az egyik leggyakoribb kiindulási pont a molekulák xyz szerkezete, mivel így lehet tudni az egyes atomok egymáshoz viszonyított elhelyezkedését a térben.

2.1.1. SMILES stringek [1]

A SMILES[1] stringek az egy formális nyel, ami kémiai struktúrákat ír le ASCII[2] karakterek segítségével tömör formában. A SMILES "nyelv" szabályai:

- Az atomokat atomszimbólumok jelölik.
- Szögletes zárójelben meg kell adni azokat az atomokat, amik nem a szerves részhalmazba[3] tartoznak.
- A hidrogén atomot nem jelölik, azt el lehet hagyni.
- Az aromás atomokat kis betűvel jelölik.
- A szomszédos atomok egymás mellett helyezkednek el.
- Az egyes kötéseket a "-" jelöli, de nem szükséges kitenni. A kettes kötéseket a "=", a hármaskötéseket a "#", aromás kötéseket a ":" jelöli, de az aromás kötések jelét sem szükséges alkalmazni.
- Az egyes ágakat zárójelbe kell tenni.
- A ciklikus struktúrákat (gyűrűket) úgy kell ábrázolni, hogy a megszakítjuk a gyűrűt, a kezdő atom és a záró atom mellé egy számot írunk (például c1ccccc1 benzol).
- Ha egy atomnak ismeretlen a szimbóluma, akkor a "*" segítségével lehet ábrázolni.

A SMILES string ábrázolás egyértelmű, mivel egy SMILES stringgel ábrázolt karaktersorozat egyértelműen meghatároz egy szerkezetet. Viszont egy szerkezetet többféleképpen is le lehet írni, így nem egyedi (például az etanolt ábrázolhatjuk CCO, OCC, C(O)C, mind a három jelölés megfelelő). Az ábrázoló algoritmusok garantálják a kanonikus reprezentációt azzal, hogy egy adott struktúrát mindig ugyan azzal a SMILES striggel ábrázol.



(a) CCN(CC)CC (b) O1CCCCC1N1CCCCC1

2.2. ábra. SMILES stringek

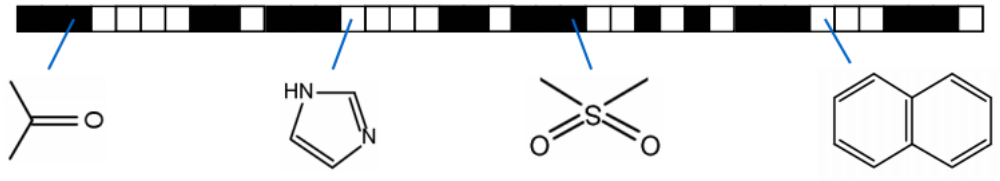
2.1.2. Fingerprint [2]

A molekulaszerkezetek és tulajdonságainak az ábrázolása egy bit sorozattal. A kétdimenziós és a háromdimenziós jellemzők bináris értékek vektoraként vannak ábrázolva. Tehát a fingerprinteknek az egyes bit pozíciói az elemek jelenlétét vagy éppen a hiányát ábrázolják.

A bitsorozat hossza változhat, attól függ, hogy milyen típusú fingerprintről van szó. A MACCS az 166 bites, a BCI pedig 1052, de létezik többfajta ábrázolás. Lehet rövidíteni is a fingerprintek hosszát, ha tömöríteni szeretnénk az adatokat.

[1] Simplified molecular-input line-entry system
[2] American Standard Code for Information Interchange
[3] organic subset (B, C, N, O, P, S, F, Cl, Br, I)

2.3. ábra. Egy példa a fingerprintre.

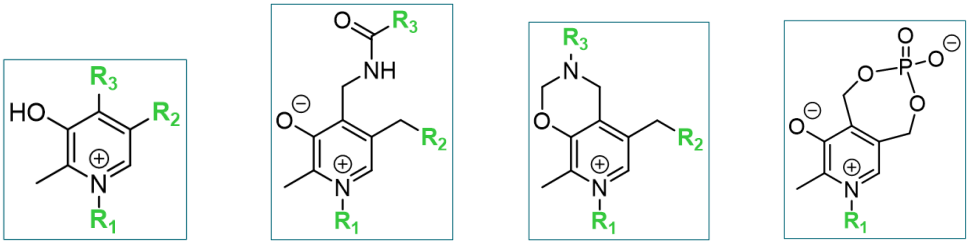


2.4. ábra. Egy példa a fingerprint részeire, generálására.

2.2. Adathalamzok

2.2.1. TTK molekulák

A Természettudományi Kutatóközpontból kapott molekulákat fel lehet osztani négy nagy csoportra (pyr1, pyr2, pyr3, pyr4), négy különböző alapvázra épülő molekulák alkotják (2.5). A négy nagy csoport tovább lehet még osztani kisebb csoportokra, ahol változnak az adott szubsztituensek. Az alapvázakra épülő szubsztituensek lehetnek különböző funkciós csoportok, vagy kisebb molekularészletek. Jelen esetben **R1** 8 fajta lehet, **R2** pedig 113. Az **R3** szubsztituensek a pyr1 rendszerre 3 csoportot jelölnek ($-COOH, -COPh, -COCF_3$), míg a pyr2 ($-Ph, -CF_3$) és pyr3 ($-Ph, -Pr$) molekulákra csupán 2-2-t.



(a) pyr1 (b) pyr2 (c) pyr3 (d) pyr4

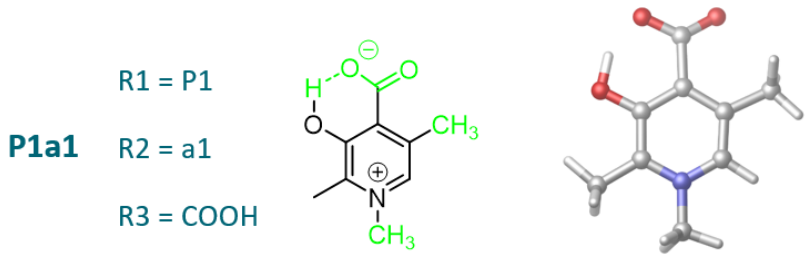
2.5. ábra. Molekula szerkezetek

A molekulaszámok az egyes csoportokban:

- **pyr1**: $8 \cdot 113 \cdot 3 = 2712$
- **pyr2**: $8 \cdot 113 \cdot 2 = 1808$
- **pyr3**: $8 \cdot 113 \cdot 2 = 1808$
- **pyr4**: 8

Viszont nincs kiszámolva az összes molekulához a redukciós potenciál értéke. Az első szettben 2580, a másodikban 1804, a harmadikban 1808 és a negyedik csoportban pedig meg van mind a 8-nak. Így összesen 6200 molekulát lehetne felhasználni a tanításhoz,

viszont a molekulák beolvasásához az RDKit [14]-et használom, amely szerint nem megfelelő az összes molekulának az elhelyezkedése, vagy éppen a kötések az atomok között. A 6200-ból 5745 redox potenciállal rendelkező molekula alkotja a tanítási adathalmazomat.



2.6. ábra. Egy példa molekula szerkezet két dimenzióban és három dimenzióban ábrázolva.

2.2.2. QM7b [3] [4] [5]

A QM7b adatkészlet a QM7 adatkészletnek kiterjesztése többfeladatos tanuláshoz kiegészítve további 13 tulajdonsággal, így már 14 tulajdonságot tartalmaz (például polarizálhatóság, ionizációs potenciál, HOMO[4] és LUMO[5] sajátértékek, gerjesztési energiák). Továbbá a molekulák száma is ki lett bővítve klóratomokat tartalmazókkal, így már összesen 7211 molekulát tartalmaz az adatkészlet.

Az adatkészlet két multidimenziós tömbből áll $X(7211 \times 23 \times 23)$ és $T(7211 \times 14)$, amelyek a bemeneteket (Coulomb-mátrixok) és a címkéket (molekuláris tulajdonságok) reprezentálják, valamint egy 14-es méretű tömbnév, amely tartalmazza a különböző tulajdonságok megnevezéseit.

2.3. MoleculeNet [6]

A MoleculeNet az egy benchmark, amit arra terveztek, hogy teszteljék a molekuláris tulajdonságok gépi tanulási módszereit. Számos adatkészletet tartalmaz, beleértve a QM7b-t (2.2.2) is. Olyan szoftvercsomag, amely számos ismert jellemzőt és már korábban javasolt algoritmust valósít meg. Minden módszer és adatkészlet a nyílt forráskódú DeepChem csomag (MIT licenc) részeként van integrálva.

A MoleculeNet több nyilvános adatbázisra épül, amit 4 nagy csoportra (2.7) lehet felosztani: kvantum mechanikai, fizikai kémiai, biofizikai és fiziológiai vagy élettani. A teljes gyűjtemény jelenleg több mint 700 000 vegyületet tartalmaz, amelyeket különböző tulajdonságokkal (például ionizációs potenciál) teszteltek. Teszteli a különböző gépi tanulási modellek teljesítményét különböző jellemzőkkel az adatkészleteken. Az eredményeket az AUC[6]-ROC[7], AUC-PRC[8], RMSE[9] és MAE[10] értékekben lehet megtekinteni.

[4] highest occupied molecular orbital
[5] lowest unoccupied molecular orbital
[6] Area Under The Curve
[7] Receiver Operating Characteristics
[8] Precision Recall Curve
[9] root mean square error
[10] mean absolute error

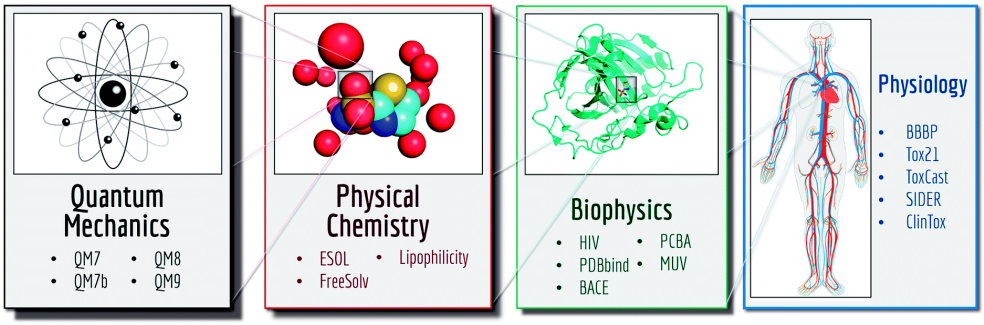
A ROC görbe a TPR[11], azaz a valódi pozitív arány és a FPR[12] hamis pozitív arány ábrázolásával jön létre különböző küszöbértékeken.

$$FPR = \frac{False\ Positive}{False\ Positive + True\ Negative} \tag{2.1}$$

$$TPR = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{2.2}$$

A pontosságot, vagy más néven a PPV[13]:

$$PPV = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{2.3}$$



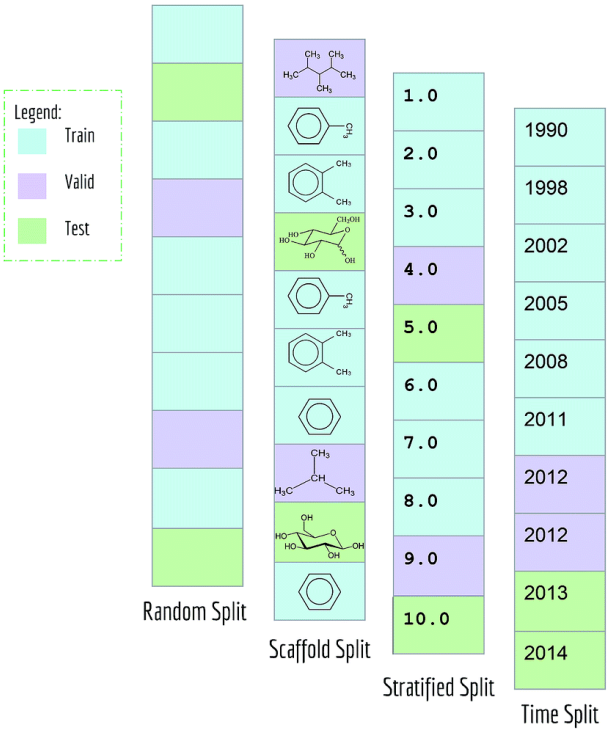
2.7. ábra. A MoleculeNet adatbázisának 4 nagy csoportja.

Az adatokat különböző módszerek alapján osztja fel 3 adathalmazra, tanítási, validációs és tesztelési halmazra, melyeknek a nagysága 80%, 10% és a tesztelési halmaz is 10%.

A négy különböző adatfelosztási módszer:

- **Véletlen felosztás** - Az adatok véletlen felosztása tanítási, validációs és tesztelési halmazba.
- **"Scaffold", struktúra alapú felosztás** - Az adatok felosztása struktúrájuk alapján tanítási, validációs és tesztelési halmazba.
- **Rétegezett véletlen felosztás** - Az adatok az adatbázisban valamilyen szempont szerint osztályozva vannak különböző csoportokba. Ez a módszer a csoportok alapján sorba rendezi növekvő címke érték szerint, majd felosztja tanítási, validációs és tesztelési halmazba.
- **Idő alapú felosztás** - Az adatok valamilyen sorrendben bekerültek az adatbázisba, a tanítási, validációs és a tesztelési halmaz kialakítását ez alapján csinálja meg.

[11] true positive rate
[12] false negative rate
[13] positive predictive value



2.8. ábra. A MoleculeNet adatbázisának a felosztása.

A MoleculeNet többfajta gépi tanulás algoritmus alapján tartalmaz modelleket melyek lehetnek hagyományos vagy gráf alapú modellek, metódusok (gráfkonvolúció, Weave modell).

3. fejezet

Módszerek

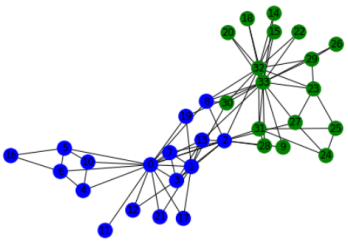
3.1. GCN

A **GCN**[1] gráf konvolúciós hálózatok alapja a gráfok és a konvolúció. A **GCN** legelterjedtebb felhasználása az osztályozási és a regressziós feladatok.

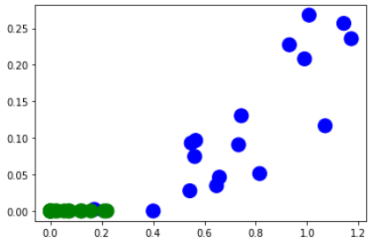
3.1.1. GCN mintapéldák

Az alapvető mintapéldák segítségével szeretném bevezetni a **GCN**-t, ezeket az alapvető mintapéldákat én is megcsináltam, azokból szeretném megmutatni az általam kapott eredményeket.

A legalapvetőbb feladat az a csomópont osztályozás, amikor a gráf csomópontjait valamilyen szempont szerint különböző osztályokba kell sorolni.



(a) A karate klub gráf reprezentációja



(b) Osztályozás eredménye 2 csoportba

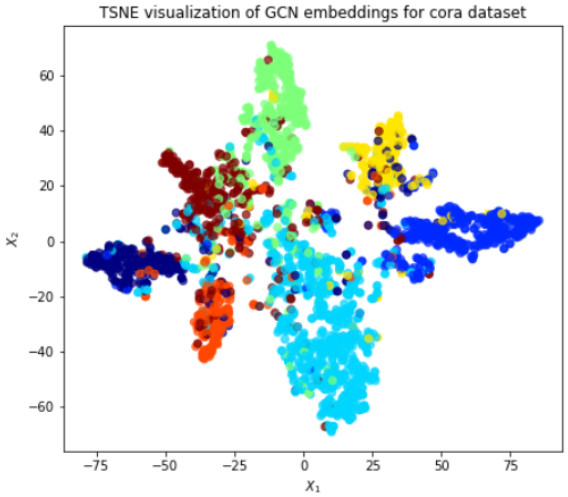
3.1. ábra. Zachary's Karate Club [15] [9]

A másik alapvető feladat pedig az él predikció, vagyis predikciót kell végezni, hogy az adott csomópontok között van-e él vagy nincs.

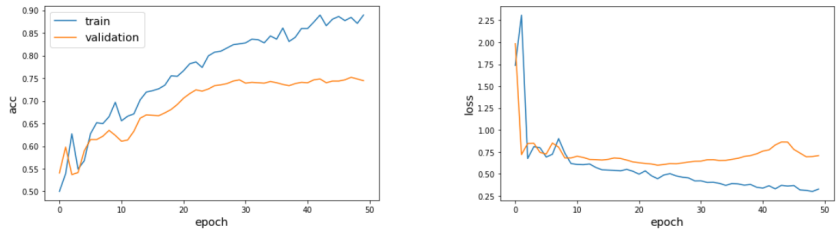
3.1.2. A GCN elmélete

Egy gráfot a csomópontjai és az élei határozzák meg, $G = (V, E)$. Több fajta gráfot lehet megkülönböztetni az éleinek a függvényében, ismerünk irányított, irányítatlan és súlyozott gráfokat. Miért jó a gráf konvolúció? Azért jó, mert a molekulákat könnyen lehet gráfként értelmezni. Az atomok azok a csomópontok, a közöttük lévő kötések pedig a gráf élei. A molekulákat egy irányítatlan gráfként kell elképzelni.

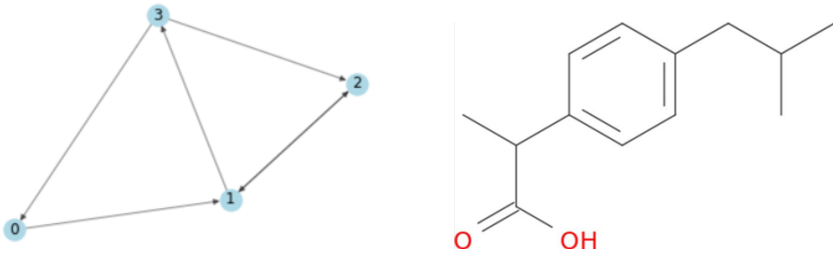
[1] GCN - Graph Convolutional Networks



3.2. ábra. Egy másik csomópont osztályozás [16]



3.3. ábra. Az él predikció tanítási eredménye [17]



3.4. ábra. Különböző gráfok

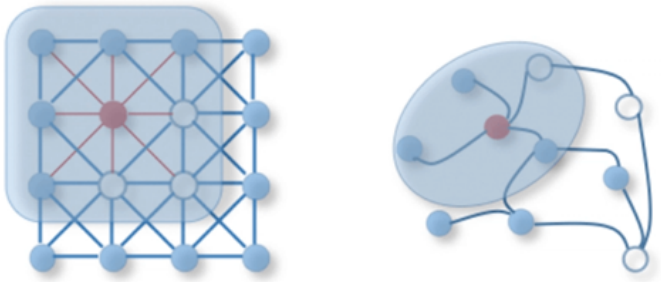
3.1.2.1. Feltételek [7]

Ahhoz, hogy a molekulákat gráfként értelmezzük bizonyos feltételeknek teljesülniük kell, nem szabad hogy a modell ezek végett másképp értelmezze a molekulákat.

- Sorrendi invariancia, a modell kimenetének invariánsnak kell lennie arra, hogy az atom és a kötés információk milyen sorrendben vannak leírva. Egy gráfot, molekulát sokféleképpen le lehet írni, sokféleképpen lehet sorba rendezni az atomokat és az éleket, de attól még ugyan az a gráf, molekula.
- Műveletek konzisztenciája, hogy ha egy gráfot, molekulát megszorzunk valamivel, vagy valamilyen műveletet végzünk rajta, akkor annak arányosnak kell maradni az eredetihez képest.

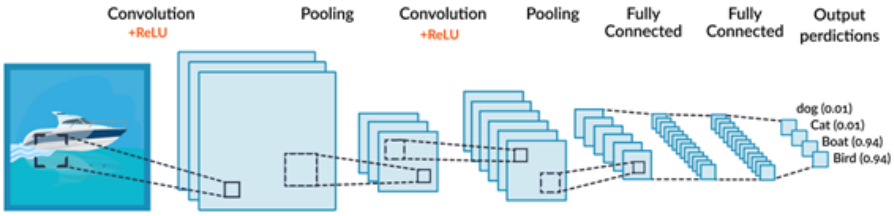
- Nem számít a csomópontok sorrendje az élek szempontjából. Ha A és B csomópont között van él, akkor B és A között is, mivel ez egy irányítatlan gráf.

3.1.2.2. Nehézség a képfeldolgozással szemben



3.5. ábra. Képek és a gráfok [18]

A képeket lehet gráfként értelmezni, mivel az egyes pixelek lehetnek a gráf csomópontjai, az egymás mellett lévő pixelek meg mintha össze lennének kötve, tehát azok lehetnek a gráf élei. Viszont a pixelek szabályosan helyezkednek el egymáshoz lépest, ha a képet gráfként értelmezzük, akkor szomszédos pixelek, ahol élek mennek, szépen egymás mellett helyezkednek el. Ezzel szemben a gráfoknak nincs meghatározott helyük, attól, hogy 2 csomópont között megy él, még nem határozza meg a helyzetüket, lehet, hogy a legtávolabb vannak, de az is lehet, hogy a legközelebb az adott molekulában.



3.6. ábra. Egymás utáni rétegek [18]

3.1.3. Matematikai háttér [8] [9]

A **GCN** az egy nagyon hatékony neurális hálózati architektúra a gráfokon történő gépi tanuláshoz. A **GCN**-nek 2 inputra van szüksége:

- X, ”feature”, tulajdonság mátrix
- A, ”adjacency”, szomszédossági mátrix

A tulajdonság mátrix az $N \times F$ nagyságú mátrix, a szomszédossági mátrix pedig $N \times N$, ahol az N az a csomópontok száma, F pedig a csomópontok tulajdonságainak a száma.

A **GCN** rejtett rétege így írható fel:

$$H^i = f(H^{i-1}, A) \tag{3.1}$$

ahol a $H^0 = X$ és az f pefig a ”propagation rule”. Tehát a következő rejtett réteget mindig az előtte lévőből és a szomszédossági mátrix segítségével kapjuk meg.

```
X
```

```
matrix([[ 0.,  0.],
        [ 1., -1.],
        [ 2., -2.],
        [ 3., -3.]])
```

(a) Tulajdonság mátrix

```
A
```

```
matrix([[0., 1., 0., 0.],
        [0., 0., 1., 1.],
        [0., 1., 0., 0.],
        [1., 0., 1., 0.]])
```

(b) Szomszédossági mátrix

3.7. ábra. GCN input mátrixok

Az egyik legegyszerűbb ”propagation rule”:

$$f(H^i, A) = \sigma(AH^iW^i) \tag{3.2}$$

ahol W^i az a súlymátrix, σ pedig egy nem lineáris aktivációs függvény, nagyon gyakran ReLU szokott lenni.

Leegyszerűsítve a ”propagation rule”-t:

$$f(X, A) = XA \tag{3.3}$$

Az egyes csomópontok ábrázolása most a szomszédos jellemzőinek összessége! Más szavakkal minden csomópont a szomszéd csomópontok jellemzőinek az összességével van jellemezve, ami baj, mert a saját jellemző nincsenek benne, ha van benne saját hurok, akkor az így elvész. Tehát akkor bele kell venni a saját jellemzőit is. Az úgy tudjuk megoldani, hogy a szomszédossági mátrixhoz hozzáadjuk az identitás mátrixot, vagyis az egység mátrixot.

$$\hat{A} = A + I \tag{3.4}$$

Ezután normalizálni kell a jellemzők, tulajdonságok ábrázolását. A normalizálást a fokmátrix segítségével lehet megoldani, pontosabban be kell szorozni az inverz fokmátrixszal a szomszédossági mátrixot.

$$f(X, A) = D^{-1}XA \tag{3.5}$$

```
D
```

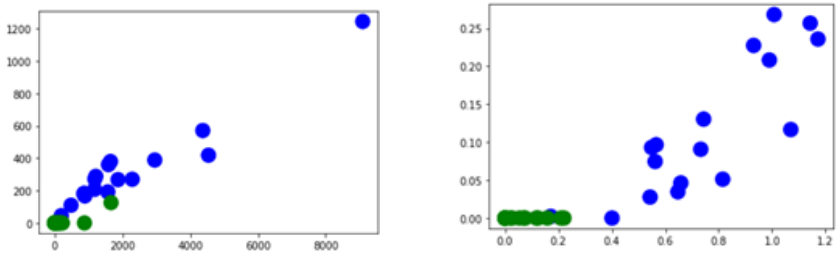
```
matrix([[1., 0., 0., 0.],
        [0., 2., 0., 0.],
        [0., 0., 1., 0.],
        [0., 0., 0., 2.]])
```

3.8. ábra. A fokmátrix szemléltetése

Mint látható a normalizálás után is meg lehet különböztetni a 2 csoportot, vagyis el lehet végezni az osztályozást, csak sokkal kisebb számokkal kell foglalkozni, ami gyorsítja a feladat elvégzését.

Tovább lehet rajta javítani, ha nem sima ”propagation rule”-t használunk, hanem egy ”spectral propagation rule”-t.

$$f(X, A) = \sigma(D^{-0.5}\hat{A}D^{-0.5}XW) \tag{3.6}$$



(a) Normalizálás előtt (b) Normalizálás után

3.2. 3DGCN [10]

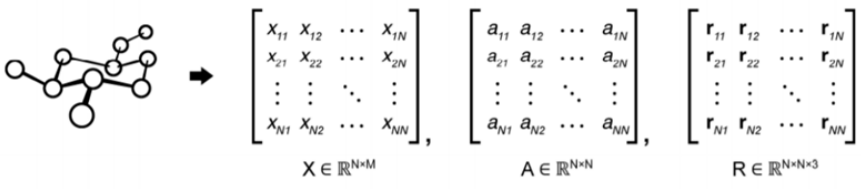
A **3DGCN** (3.2) a sima **GCN** (3.1) kibővített változata, csak nem két dimenzióban, hanem három dimenzióban. Tehát a **3DGCN** az nem más mint egy háromdimenziós gráf konvolúciós hálózat, amely képes predikciót, regressziót végezni molekuláris tulajdonságok és biokémiai aktivitásokra, 3D molekuláris gráfok alapján. Mivel 3D-s, ezért több információ áll rendelkezésére, mint egy sima 2D-s gráf ábrázolás, ezáltal jobb az általánosítóképessége. Mivel 3D-s modell, 3D-s gráfokkal dolgozik ezért invariáns a forgatásra. A forgatás invariancia az egy nagyon hasznos tulajdonsága ennek a modellnek, mert attól, hogy egy molekulát a térben elforgatunk, még ugyan azt a molekulát kapjuk meg, csak egy másik reprezentációja a térben, de a prediktált redukciós potenciál értékének attól nem szabad változnia.

3.2.1. A modell leírása

Mivel 3D-s gráfokkal dolgozik, ezért nem elég 2 mátrix a bemenetére, mint a sima **GCN**-nél, itt 3 inputra van szüksége:

- X, ”feature”, tulajdonság mátrix
- A, ”normalized adjacency”, normalizált szomszédossági mátrix
- R, relatív helyzet mátrix

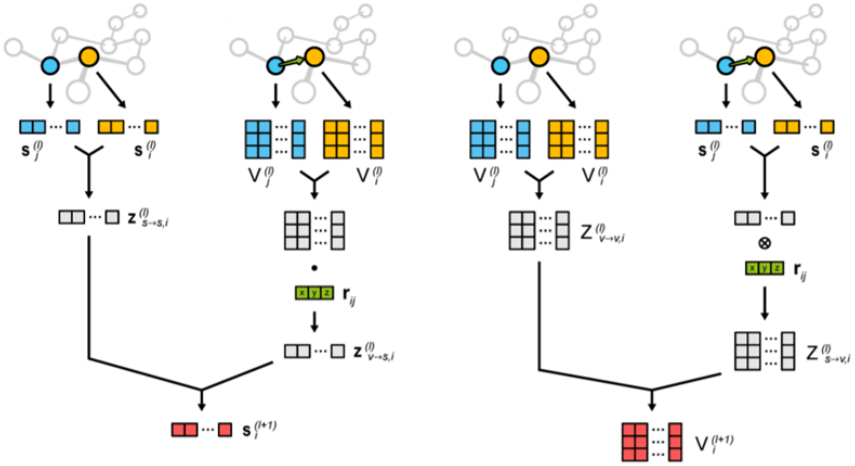
A tulajdonság mátrix $X \in R^{N \times M}$, a normalizált szomszédossági mátrix $A \in R^{N \times N}$, a relatív helyzet mátrix $R \in R^{N \times N \times 3}$, ahol az N az atomok száma, az M pedig az atom tulajdonságoknak a száma.



3.9. ábra. A **3DGCN** 3 bemeneti mátrixa

A relatív helyzet mátrix az atomok egymáshoz viszonyított helyzetét tartalmazza. Mivel nem az atomok egyéni helyzetét tartalmazza, ezért tudja biztosítani ez a modell az invarianciát.

A bemeneti 3 mátrixot tovább alakítja $s \in R^M$ skaláris jellemzővé és $V \in R^{M \times 3}$ vektor jellemzővé.



3.10. ábra. Ábrázolások, reprezentációk és műveletek a háromdimenziósság kezelésére a **3DGCN**-ben

A (3.10). ábrán látható a skaláris és a vektor jellemzők közötti négy művelet a tulajdonságok frissítése során. A központi csomópont az ábrán narancssárgával van jelölve, a szomszédos csomópontja kékkel, szürkével van jelölve ilyen közbenső tulajdonságok, zölddel a relatív pozíció vektor és pirossal a következő szintű tulajdonságok. A közbenső tulajdonságot a központi csomópont és a vele szomszédos csomópontokból származó skaláris vagy vektor jellemzők lineáris kombinációiból lehet megkapni. Akkor amikor a skalárból szeretnék áttárni vektorra, vagy fordítva, akkor kell használni a relatív pozícióvektort, hogy a dimenziók rendben legyenek. Amikor skalárból akarunk áttérni vektorba, akkor a skaláris szorzást kell használni, amikor pedig a vektorból skalárba akarunk áttérni, akkor pedig a tenzor szorzást, vektoriális szorzást kell használni. Így lehet előállítani a következő szintű tulajdonságokat.

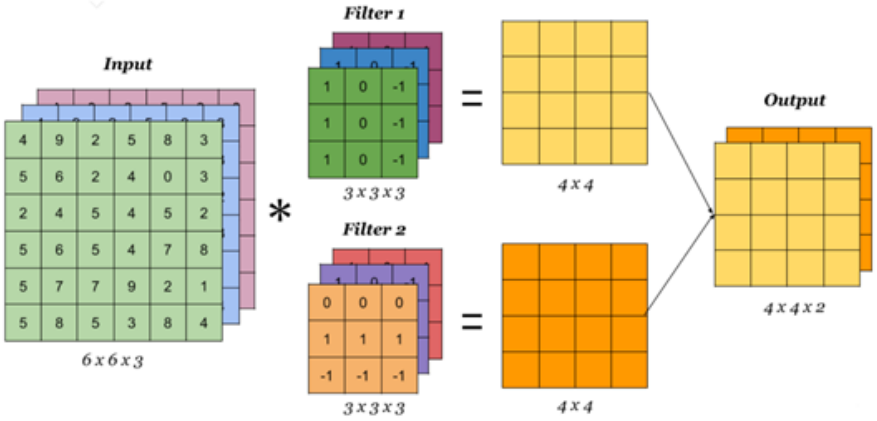
A skaláris kimenetekhez aktiválásához ReLU aktivációs függvényt használ a modell, a vektor kimenetekhez pedig tanh aktivációs függvényt használ.

3.2.2. A 3DGCN moduljai

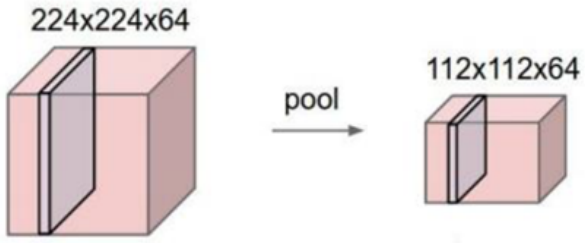
A **3DGCN** 3 modulból áll:

- konvolúciós, ”convolutional” rétegből
- jellegzetesség összesítő, ”pooling” rétegből
- teljesen összekapcsolt, ”fully connected” rétegből

Mind a 3 típusú réteget másra használják, más a feladata az adott rétegeknek. A konvolúciós rétegben a bemeneti mátrixokból előállított közbenső tulajdonságokat szűrökkel végig konvolválja, azaz skalárszorzást végez a mátrix minden egyes részével, elvégzi a konvolúciót. A konvolúciós rétegek a jellemzők kiemelésére szolgálnak. A konvolúciós réteg után a pooling réteg következik. A **3DGCN** esetén 2 fajta poolingot tud megkülönböztetni a max és a sum poolingot. A pooling réteg feladata a, hogy csökkentse a reprezentáció méretét, ezáltal kezelhetőbbé teszi. A teljesen összekapcsolt rétegek a modell utolsó rétegeiben található meg, amik elvégzik az osztályozást, ez esetben a regressziót.



3.11. ábra. A konvolúciós rétegek szemléltetéses szűrők segítségével



3.12. ábra. A pooling rétegek szemléltetéses

3.3. A 3DGCN kibővítései

Nagyon jó az alapkoncepciója a 3DGCN-nek, de mindig lehet egy adott témakört tovább gondolni, kibővíteni. Maga a teszteléshez, hiperparaméter optimalizáláshoz is ki lehetett bővíteni plusz funkciókkal. A rétegek elrendeződésével meg kaphatunk egy új koncepciót, egy újfajta megközelítést.

3.3.1. A tanítási, validációs és teszthalmaz nagyságának beállítása

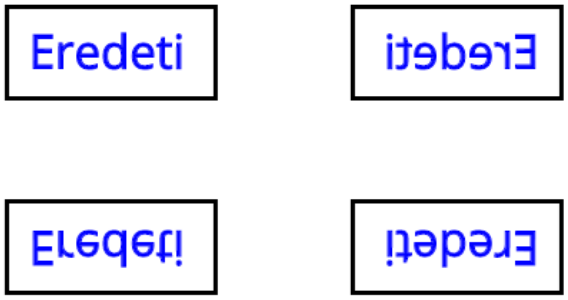
A 3DGCN alapból 3 részre osztja fel az adathalmazt, tanítási 80%, validációs 10% és a tesztelési halmaz is 10%. Azonban nem mindig ez a legjobb felosztás, van hogy másképp kellene felosztani. Ha nagyon nagy az adathalmazunk, akkor lehet, hogy több adatot szeretnék hagyni tesztelésre vagy validációra mint 10%. Ha pedig kevés adatunk van, akkor általában augmentálni (3.3.2) szoktuk az adathalmazt, mert szükség van egy bizonyos mennyiségű adatra, hogy elvégezhessük a tanítást.

Ezzel a kibővítéssel, ha a Trainer inicializáláskor nem adjuk meg, hogy milyen nagy legyen a három halmaz nagysága, akkor az alap eset lesz beállítva, vagyis a tanítási 80%, a validációs 10% és a tesztelési is 10%. Ha pedig megadjuk, akkor annak megfelelően fogja felosztani az adathalmazt. Ha úgy szeretnénk felosztani, hogy csak 50% legyen a tanítási halmaz, 20% a validáció és 30% a tesztelési halmaz, akkor a *split* paramétert hamisra kell állítani, igaz alap esetben ez hamis, ezt egy későbbi bővítés végett (3.3.3) vezettem be, a *valid_size*-t 0.2-re, a *test_size*-t pedig 0.3-ra. A tanítási halmazt már tudni fogja, hogy 0.5-re kell beállítani.

A tanítási halmaz csökkentésének a következményeit az eredményeknél (5.3) szeretném ismertetni.

3.3.2. Augmentálás elvégzése

Ha kevés adattal rendelkezünk, akkor gyakran ki kell bővíteni az adathalmazt. Az adathalmaz bővítésének az egyik leggyakoribb módja az az, ha augmentáljuk az adatokat. Ami annyit tesz, hogy a meglévő adatokat valamilyen szempont alapján elforgatjuk, torzítjuk, tükrözzük és az így keletkezett adatokkal kibővítjük az eredeti adathalmazt. Képeknél a legegyszerűbb módja az augmentálásnak, ha elforgatjuk, vagy tükrözzük a képeket. Ha egy képen elvégzünk egy horizontális tükrözést, majd utána az eredetin és a tükrözöttön is egy vertikális tükrözést, akkor mindjárt meg négyszereztük az adathalmaz nagyságát (3.13).



3.13. ábra. Egy kép agumentálása

A gráfoknál, molekuláknál is a forgatás az egyik legjobb módszer az adatok dúsítására, augmentálására. Kibővítettem egy funkcióval, amit tanítás előtt lehet elvégezni, ha ezt nem csinálja meg a felhasználó, akkor csak az eredeti adatokkal fogja elvégezni a modell a tanítást. A felhasználónak meg kell adni az eredeti adathalmazt, majd azt hogy mi alapján akarja elforgatni a molekulákat, lehet véletlenszerű forgatás vagy paraméterezett forgatás. A paraméterezett forgatást úgy kell érteni, hogy a felhasználó adja meg a forgatás feltételeit, azt hogy melyik tengely mentén és hogy milyen szögben (x, y, z tengely szerinti forgatás a megadott szög, szögek alapján).

Ezáltal tudjuk ellenőrizni, hogy invariáns-e a forgatásra. Természetesen óvatosan kell augmentálni az adatokat, nem lehet csak úgy összevissza. Ha az egész adathalmazt egyben augmentáljuk, akkor gondban leszünk mikor felosztjuk tanítási, validációs és tesztelési halmazra. Az nem jó, ha egy konkrét molekulánk valamelyik augemtált változata bekerül a tanítási, validációs és a teszt halmazba is, mivel akkor nincs jól szétválasztva egymástól a három adathalmaz. Elveszti a szerepét az a koncepció, hogy a tesztelési adatok ne legyenek ismertek tanítás közben. Ezért sokkal jobb, ha először szétosztjuk 3 halmazra az eredi adathalmazt, külön augmentáljuk őket, majd egy általam írt szkript segítségével összefésüljük a 3 adathalmazt. Ha így használjuk az augmentálást, akkor a (3.3.3) kibővítés segítségével konkréten meg tudjuk adni, hogy melyik adatok szerepeljenek a tanítási, validációs és tesztelési adathalmazban.

3.3.3. Konkrét tanítási, validációs és teszt halmaz megadása

Sokszor nem jó, ha véletlenszerűen kerülnek az adatok a tanítási, validációs és tesztelési adathalmazba, van olyan eset, amikor jobb, ha mi állítjuk be az adathalmazokat. Természetesen általában jobb véletlenszerűen szétosztani az adatokat, mert ha mi állítjuk be, akkor az már befolyásolhatja a tanítás eredményét, persze van hogy pont az a célunk. Ha például egy osztályozást végzünk 3 osztályra, nem jó úgy beállítani az adathalmazt, hogy a tanítási halmazba kerül az egyik osztály, a validációsba a másik, a tesztelésibe pedig a

harmadik osztály. Ha így osztanánk fel, akkor a modellünk csak az egyik osztályra tanulna ra, a többinél pedig nagyon rossz lenne a hatásfoka. Viszont, ha augmentáljuk az adatokat, akkor nem jó egyben augmentálni az egész adathalmazt, mert ha csak az augmentálás után osztjuk fel az adathalmazt és nem figyelünk oda a felosztásnál, akkor az eredi kép és az augmentált változataiból kerülhet mind a három halmazba, ami nem jó.

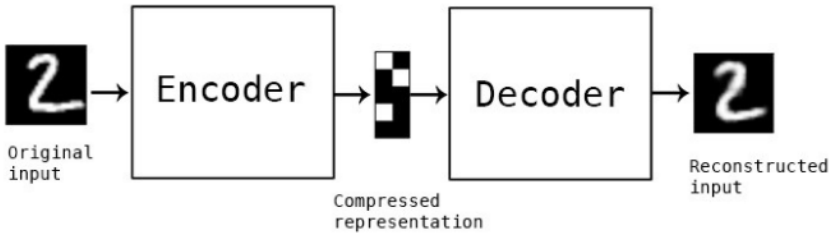
Ezzel a kibővítéssel, ha a Trainer inicializáláskor nem adjuk meg, hogy melyik halmazba mennyi adat legyen, akkor az alap eset lesz beállítva, vagyis a tanítási 80%, a validációs 10% és a tesztelési is 10%. Ha pedig megadjuk, akkor annak megfelelően fogja szétosztani az adatokat az adathalmazból. Legyen 3000 adatunk és azt szeretnénk elérni, hogy a tanítási halmazban pontosan 2000 adat legyen, a validációsban 400 adat és a tesztelési halmazban pedig a maradék 600, akkor a következő paramétereket be kell állítanunk. A *split* paramétert igazra kell állítani, alap esetben hamis, a *valid_size*-t 400-ra, a *test_size*-t pedig 600-ra. A tanítási halmazt már tudni fogja, hogy 2000-re kell beállítani.

3.4. 3DGCN és autoenkóder

Ha változtatunk a modell belső szerkezetén, a konvolúciós rétegegek nagyságán akkor kaphatunk egy újfajta megközelítést. Megpróbáltam a 3DGCN-be beleépíteni egy enkóder és egy dekóder részt, ami az autoenkódernek az alapja. Ezzel azt szerettem volna elérni, hogy jobb legyen az általánosítóképessége a modellnek.

3.4.1. Autoenkóder [11]

Az autoenkóder az egy fajtája a mesterséges neurális hálóknak, pontosabban egy felügyelet nélküli[2] mesterséges neurális hálózat. Amely rátanul hogyan lehet az adatokat hatékonyan tömöríteni, kódolni, majd rátanul, hogy ebből a csökkentett ábrázolásból hogyan rekonstruálja az eredeti bemenethez lehető legközelebb eső ábrázolást, adatokat. Az autoenkóder szerkezetének a kialakításával, a csökkentett dimenziókkal rátanul hogy hogyan hagyhatja figyelmen kívül az adatok zaját.



3.14. ábra. Autoenkóder az MNIST[3]adatbázison

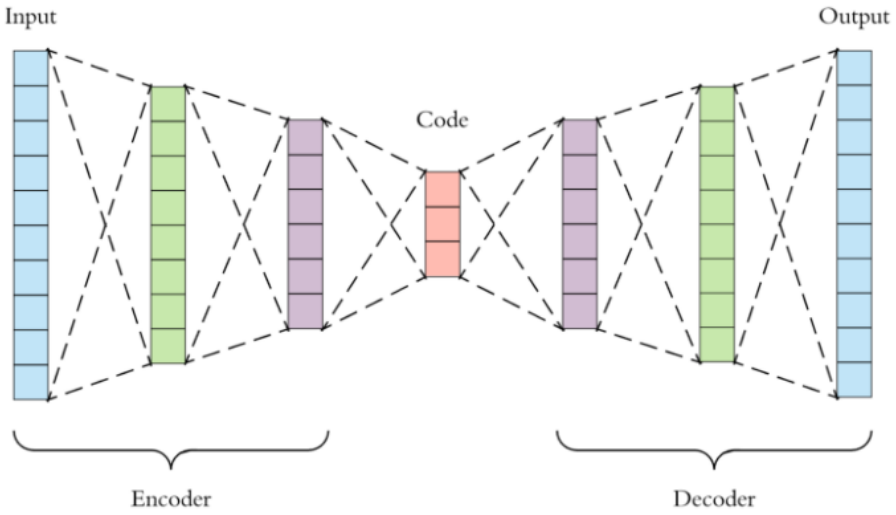
Az autoenkóder komponensei:

- *Enkóder* - Az enkóder feladata, hogy csökkentse a bemeneti adat méretei, rátanuljon hogyan tömörítse az információkat. Folyamatosan csökkenti a rétegek nagyságát.
- "*Bottleneck*" - Ez az enkóder utolsó rétege, tartalmazza a bemeneti adatok tömörített változatát, ez a lehető legkisebb.

[2]unsupervised artificial neural network
[3]MNIST[19] adatbázis - kézzel írott számjegyek

- *Dekóder* - A dekóder folyamatosan növeli szimmetrikusan az enkóderrel a rétegek nagyságát, amíg az eredeti nagysághoz nem ér. Itt a modell megtanulja, hogy hogyan lehet rekonstruálni tömörebb információból az eredetihez hasonló ábrázolást.

Az autoenkódernél nagyon fontos az újjáépítési veszteség[4], ez egy mérőszám, ami a dekódert jellemzi, hogy mennyire jól teljesít, milyen közel van az eredeti adat a keletkezetthez.



3.15. ábra. Az autoenkóder architektúrája [20]

3.4.2. 3DGCN AE

Megpróbáltam a 3DGCN-t kiegészíteni egy olyan struktúrával mintha autoenkóder lenne. Ezzel a jobb általánosítóképességét szerettem volna javítani. Ha a tömörebb információra rá tud tanulni, akkor az újabb típusú molekulákra (gráfokra) jobb eredményt tud adni. Egy neurális hálózatnál fontos, hogy jó legyen az általánosítóképessége.

Az autoenkóder jellegét az adja, hogy a konvolúciós rétegeknél a skaláris és a vektoriális jellemzők segítségével összeállított rétegek egy folyamatos filternagyság csökkenéssel, majd egy növelésből állnak. A paraméterül megadott units_conv értéket folyamatosan felezi, ez az enkóder rész, a másik paraméterül adott num_layers érték nagyságáig. Majd ha ezt elérte, utána folyamatosan növeli a filter mélységét amíg az eredeti filter nagysághoz nem ér.

Ahhoz, hogy ezt a modellt használjuk, akkor a trainer.fit() metódusában a modell paraméternek meg kell adni, hogy az autoenkóderes *model_3DGCN_AE* modellt használja.

[4]Reconstruction Loss

4. fejezet

Vizsgálatok, számítások menete

4.1. Redukciós potenciál

Különböző kémia reakciók léteznek. Egy kémiai reakciót redoxifolyamatoknak vagy redoxireakcióknak nevezzük, ha az oxidációs szám megváltozásával járnak, az oxidációs szám csökkenhet vagy nőhet is. Ezekben a folyamatokban két reakciópartnert különböztetünk meg, az egyik felvesz (**oxidálószer** [21]), a másik pedig veszít (**redukálószer** [22]), más szóval lead elektronokat. Ha az oxidációs száma nő, akkor elektront add le, tehát oxidálódik. Viszont ha az oxidációs száma csökken, akkor elektront vesz fel, ebből következik, hogy redukálódik.

A redox[1] reakciók legfontosabb paramétere a redoxpotenciál, amit az angol nyelvközegben ORP[2]-nek neveznek). Egy rendszert redoxi rendszernek nevezünk abban az esetben, ha egy közegben megtalálható a redukáló és az oxidáló anyag is egyszerre [23]. Melynek az egyszerűsített formája:

$$Red = Ox + e \tag{4.1}$$

Ha kiszeretnénk számolni egy molekulának a redukciós potenciál értékét, akkor viszonylag sok számítást kell elvégeznünk, mert a pontos számolása az egy hosszadalmas folyamat. Szeretném ismertetni azt a folyamatot, amikor egy standard képlet alkalmazásával ki lehet számolni egy molekulának a redukciós potenciál értékét, pontosabban a molekulák 1 elektronos redukciós potenciálját.

A jelen munkámat együttműködésben végeztem a TTK Elméleti Kémiai Osztály munkatársaival. A kvantumkémiai számolásokat, teszteléseket Hamza Andrea [24] végezte, ő segített megérteni a dolgozatom alapjául szolgáló adatok kémiai hátterét. A redukciós potenciál kémiai számolása a Roth és társai [25] által publikált tanulmányban szereplő képlet alapján történt:

$$\Delta E_{1/2}^{calc} = -\frac{\Delta G_{calc}}{n_e F} + \Delta E_{1/2}^{ref} \tag{4.2}$$

$$\Delta G_{calc} = G(anion\ gyok) - G(semleges) \tag{4.3}$$

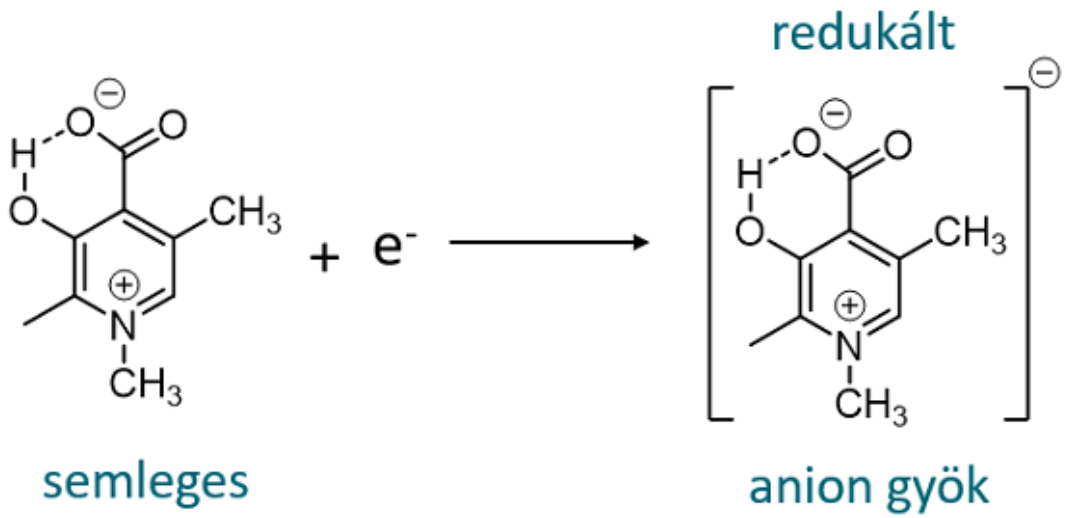
A redukciós potenciál képlete (4.2) több mindentől függ. A termék és a reaktáns szabad entalpiájától (ΔG_{calc}), amit a (4.3) képlet alapján számolunk, továbbá függ egy referencia értékétől ($\Delta E_{1/2}^{ref}$) is, ami az elektróda jellemzője. Az elektronok számát a n_e jelöli, esetünkben 1, mivel 1 elektronos redukciós potenciál, F pedig a Faraday állandó. A víz közeget tekintettük ebben az esetben referenciának. A redukció (4.1) ebben az esetben 1 elektronos. Mivel az adott komponensünk, vegyületünk felvesz egy elektront, így ezáltal

[1] Redukciós-oxidációs

[2] ORP - Oxidation Reduction Potential

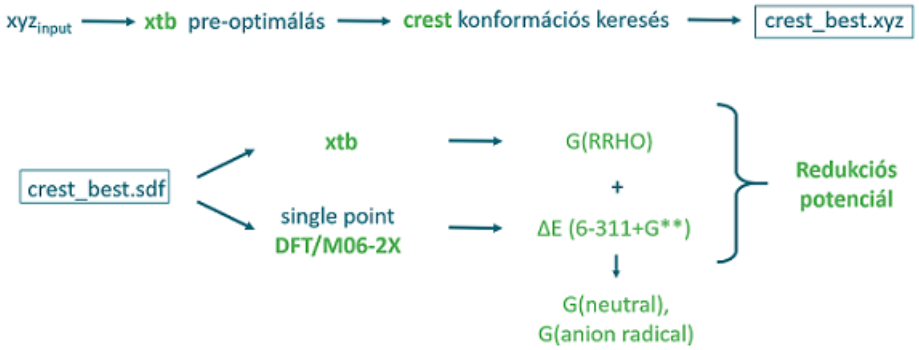
redukálódik. Amennyiben a kiindulási komponensünk semleges volt, akkor gyök anion keletkezik. Vagyis adott a vegyület, a komponensünk, ami felvesz egy elektront, ezáltal redukálódik. Így jön létre a gyök anion, amennyiben a kiindulási komponens semleges volt. Viszont, ha a kiindulási anyag töltéssel rendelkezik, akkor ennek megfelelően változik a termék töltése is. Például egy anion kiindulási anyagból egy kétszeresen negatív töltésű[3] gyök keletkezik. A reakciónak a redukált formája a gyök anion. A redukciós potenciál (4.2) meghatározásához szükségünk van még a reakció szabad entalpiájára.

Kis számú és kis méretű molekulák esetében a szabad entalpia pontos kvantumkémiai számolása könnyen elvégezhető, azonban sok molekula esetében ez már nagyon időigényessé válik. Ezért szükséges volt egy olyan munkafolyamat kidolgozása amely pontos, de ugyanakkor viszonylag gyorsan add jó eredményt.



4.1. ábra. A redukció reakciója

4.1.1. Számolási protokoll



4.2. ábra. A számolási protokoll.

Megfelelő módszer kell nagy számú molekula redukciós potenciál számolásához, egyszerre kell gyorsnak és megfelelően pontosnak lennie, mivel az idő az egy nagyon fontos

[3] -2

tényező, ugyanakkor kellően hatékonynak kell lennie. Alapos tesztelések alapján dolgozta ki Hamza Andrea [24] a *cxMs*[4]-nek elnevezett munkafolyamatot, ami alkalmas a feladat elvégzésére. A munkafolyamat elnevezése magába foglalja azokat a kvantumkémiai számításos módszereket, amelyek részei a kidolgozott munkafolyamatnak. Ennek megfelelően: x - xtb program, c - crest program, M062X a DFT (Density Functional Theory) számolásokban használt funkcionál. A "solvent" kulcsszó ebben az esetben arra utal, hogy a számolási folyamatok mindig implicit oldószer figyelembevételével történtek.

A kiindulási geometriákat xyz_{input} optimalizálni[5] kell, majd egy konformációs keresést kell végrehajtani. A konformációs keresés nagyon fontos tényező a módszerben, mivel számtalan forgási, szimmetriai művelet végbemehet egy molekula szerkezetében. Ebből a számtalan térbeli elhelyezésből ki kell választani a legkedvezőbbet, energetikai szempontból a legmélyebb konformert. Maga a pre-optimalizást a *xtb*[6] segítségével lehet elvégezni a konformációs analízist pedig a *crest*[7] segítségével. Miután megtaláltuk a legmélyebb konformert, a legideálisabbat, akkor ki lehet számolni a redukciós potenciált minden molekulára, a kiindulási anyagra és a redukáltra is ki kell számolni:

$$G = E_{DFT} + \Delta G_S \tag{4.4}$$

ahol E_{DFT} a legmélyebb konformerre számolt DFT szintű elektronikus energia, ΔG_S pedig az xtb-vel származtatott entrópia korrekció. Így már minden szükséges tényező előállt a végeredményhez, amiket a (4.2) képletbe behelyettesítve megkapjuk a redukciós potenciál értékeket.

4.2. Fájl konverzió, xyz, sdf

Nagyon fontos az adatok megfelelő előkészítése, átnézése, bizonyos szempontok szerinti kiválogatása. Egy molekulát többféle képen lehet ábrázolni, sok fajta reprezentáció létezik, melyekből ismertettem egy párat a (2.1) fejezetben. A különböző reprezentációk különböző információkat tartalmazhatnak az adott molekuláról. Egyes reprezentációk több információval bírnak a többinél, de attól még ugyan azt a molekulát reprezentálják. Valamikor elég a kevés információ rendelkezésre állása, de van olyan eset is, amikor elengedhetetlen a sok információ. Nem szükségszerű, de általában a több információ birtokában, jobb eredményeket tudunk elérni. Ebből kifolyólag a három dimenzió több információt hordoz magában, mint a kétdimenziós molekula, mivel háromdimenziós szerkezettel rendelkezik. Egy molekulát három dimenzióban leggyakrabban az xyz, azaz (4.2.1) kiterjesztésben szokták ábrázolni, mivel tartalmazza az atomok térbeli koordinátáit, viszont nem tartalmazza a molekulák kötéseit, ami gráf szempontjából nagy probléma. Ahhoz, hogy rendelkezésünkre álljon az az információ is, hogy melyik atom melyikkel van összekötve, illetve a kötés jellege is (pl. egyes kötés, kettős kötés, vagy aromás) a legalkalmasabb az sdf (4.2.2) fájl formátumot használjuk.

Szeretném a piridin C_5H_5N molekula segítségével rámutatni a különbségre az xyz és az sdf fájl között.

4.2.1. xyz

Nem létezik hivatalos szabvány az **xyz** fájlok számára, viszont ezek általában ugyanolyan típusú adatokat tartalmaznak [26]. A különböző kémiai programok (például a Gaussi-

[4] crest-xtb-M062X-solvent
[5] pre-optimalizálni
[6] Semiempirical Extended Tight-Binding Program Package
[7] Conformer-Rotamer Ensemble Sampling Tool

```
11

C        -0.180226841      0.360945118     -1.120304970
C        -0.180226841      1.559292118     -0.407860970
C        -0.180226841      1.503191118      0.986935030
N        -0.180226841      0.360945118      1.29018350
C        -0.180226841     -0.781300882      0.986935030
C        -0.180226841     -0.837401882     -0.407860970
H        -0.180226841      0.360945118     -2.206546970
H        -0.180226841      2.517950118     -0.917077970
H        -0.180226841      2.421289118      1.572099030
H        -0.180226841     -1.699398882      1.572099030
H        -0.180226841     -1.796059882     -0.917077970
```

4.3. ábra. A piridin molekula xyz kiterjesztésben.

```
 OpenBabel09012008533D

 11 11  0  0  0  0  0  0  0  0999 V2000
   -0.1802    0.3609   -1.1203 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802    1.5593   -0.4079 C   0  0  0  0  0  3  0  0  0  0  0  0
   -0.1802    1.5032    0.9869 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802    0.3609    1.2902 N   0  0  0  0  0  4  0  0  0  0  0  0
   -0.1802   -0.7813    0.9869 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802   -0.8374   -0.4079 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802    0.3609   -2.2065 H   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802    2.5180   -0.9171 H   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802    2.4213    1.5721 H   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802   -1.6994    1.5721 H   0  0  0  0  0  0  0  0  0  0  0  0
   -0.1802   -1.7961   -0.9171 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  2  1  0  0  0  0
  1  6  2  0  0  0  0
  2  3  1  0  0  0  0
  3  4  2  0  0  0  0
  3  9  1  0  0  0  0
  5  4  2  0  0  0  0
  5 10  1  0  0  0  0
  6  5  1  0  0  0  0
  7  1  1  0  0  0  0
  8  2  1  0  0  0  0
 11  6  1  0  0  0  0
M  END
$$$$
```

4.4. ábra. A piridin molekula sdf kiterjesztésben.

an, ORCA, Turbomole) által használt fájlok tartalmaznak egy molekula leírást, beleértve az atomok számát, az elemszimbólumot és az xyz koordinátákat szöveges formátumban tárolva.

4.2.2. sdf

Az SDF fájl az egy kémiai adatfájl-formátum, amely kifejezetten strukturális információk tárolására szolgál, mely az MDL által lett kifejlesztve. Az ”SDF”[8] jelentése struktúra-adatfájlok, amik tulajdonképpen beburkolják a molfile (MDL Molfile) formátumot. Egy fájl tartalmazhat több molekulát is, nem csak egyet, ebben az esetben a molekulákat négy dollárjelből álló vonal határol ($$$$). Az SDF formátum jellemzője, hogy képes társított adatokat befogadni [27].

[8] structure-data file

Az SDF fájl tartalmaz egy négy soros fejrészt[9], ahova kommenteket, megjegyzéseket lehet írni az adott molekuláról, általában tartalmazza, hogy melyik program által lett generálva. Megtalálható benne a molekula egyes atomjainak nem csak a neve és az x, y, z koordinátái, hanem egy csatlakozási táblázat is, melyik atom melyik másik atommal van összeköttetésben. Látszik, hogy az SDF fájl sokkal több információt tartalmaz mint egy sima xyz 4.2.1 fájl.

De miért jobb számomra az SDF fájl formátum? Mivel vannak csomópontok (atomok) és vannak élek (atomok közötti kötések), így már gráfként is lehet értelmezni a molekulákat.

Az xyz fájlból az SDF fájlba a konverziót[10] az OpenBabel [28] program segítségével hajtottam végre. Az OpenBabel nagyon sok fajta fájlformátumot felismer és engedélyezi a fájlok átkonvertálását más kiterjesztésbe. Ez egy eléggé ismert konvertáló program, jó referenciákkal, viszont ellenőrzést mindig kell csinálni. A konverziót követően ellenőriztem, hogy minden fájlra sikeres volt-e konverzió. A molekula beolvasására nagyon gyakran az RDKit modult [14] használják, python környezetben egy elterjedt modul molekulák kezelésére. A 3DGCN-nél szükséges, hogy az sdf fájl target-ként tartalmazza a redukciós potenciál értékeket is. Alap esetben az xyz fájl nem tartalmazza ezeket, így a konverzió során nem kerülnek bele az SDF fájlba sem. Az egyes molekulák potenciálértékeit egy külön dat kiterjesztésű fájlban kaptam meg sorszámozva a molekulák szerint. A dat fájlt átírtam egy csv formátumú fájlba, mivel a csv kiterjesztés jól támogatott python környezetben. Egy általam írt python script ebből a csv fájlból beolvassa a megfelelő sorszámú molekula redukciós potenciál értékét, nem feltétlenül van minden molekulának kiszámolva a redukciós potenciál értéke, ezeket ki kell szűrni, majd beolvassa az átkonvertált SDF fájlt és a megfelelő helyre beleírja target-ként a redukciós értéket.

4.3. Hiperparaméterek optimalizálása

Nem csak az adatok előkészítése fontos, hanem a hiperparaméterek optimalizálása is elkerülhetetlen a neurális hálók megfelelő működéséhez. Elengedhetetlen a jól előkészített adatok a jó eredményhez, mivel lehet egy nagyon jó rendszerünk, de rossz, nem konzisztens adatokkal nem tudunk jó eredményt elérni. Ugyan ez a helyzet a nem megfelelő paraméter érték beállításokkal is.

Nem biztos, hogy ugyan olyan paraméter értékek kellenek ugyan ahhoz a modellhez, ha mások az adatok, vagy azonosak a tanítási adatok, de más a rétegszerkezet a modellen belül. Létezik többfajta módszer a paraméterek finomhangolására, lehet véletlenszerűen választani különböző paramétereket, de az nem a legjobb módszer és lehet szisztematikusan választani paramétereket. Minél több esetet nézünk még, annál nagyobb valószínűséggel találunk megfelelő paramétereket. Olyan mintha egy ”brute force” optimalizálást végeznénk, természetesen az összes eshetőséget nem lehet kipróbálni, tesztelni, de ha okosan választunk tesztelni való paramétereket akkor leszűkíthetjük a keresés halmazát viszonylag jó eredménnyel.

4.3.1. Hiperparaméterek optimalizálás folyamata

Az adathalmazt beolvastam véletlenszerűen összekevertem majd kiválasztottam belőle 10%-ot amin optimalizáltam a paramétereket. Mivel véletlenszerűen választottam ki 10%-ot az adathalmazból, így kellően különböző adatokat tartalmazott. Azért 10%, mert ha sok esetet-re nézzük meg az optimalizálást, akkor az azért eltart egy ideig. 5 Paraméterre néztem meg az optimalizálást:

[9] header block
[10] xzy -> SDF

- **batch** $= [8, 16, 32]$ - A batch nagysága.
- **units_conv** $= [32, 64, 128]$ - A konvolúciós rétegekben a filter nagysága, milyen legyen a mélysége.
- **units_dense** $= [32, 64, 128]$ - A dense rétegekben a filter nagysága, milyen legyen a mélysége.
- **num_layers** $= [2, 3, 4, 5]$, A konvolúciós rétegek száma, természetesen, 1 réteg alatt azt értem, amibe bele tartozok a skaláris jellemzőkből a vektor jellemzők képzése és fordítva, amit a (3.2) fejezetben ismertettem.
- **pooling** $= [$"sum","max"$]$ - A pooling típus, lehet max vagy sum pooling.

Ha csak ezt az 5 paramétert nézzük, $3 \cdot 3 \cdot 3 \cdot 4 \cdot 2 = 216$ különböző eset. Viszont, hogy megőrizzük a konzisztenciát a tanítást nem jó úgy elvégezni, hogy az adatokat mindig újrarendezzük, mindig ugyan azoknak kell lenniük a tanítási, validációs és tesztelési halmaznak. Itt jól jön az egyik kibővítés (3.3). Maga a tanítást sem kell teljesen elvégezni, csak mindig ugyan annyi mennyiségű epoch-ig kell, az optimalizálás során 5 epochot választottam. A TTK-s adathalmaz 5745 molekulát tartalmaz, ennek a 10%-a 574, az 574 molekulát véletlenszerűen választottam ki, melyből 460 került a tanítási halmazba 57 a validációs és 57 a tesztelési halmazba, de mindig ugyan azok.

4.1. táblázat. Egy pár példa a 3DGCN optimalizálására

MAE	**batch**	**units_conv**	**units_dense**	**num_layers**	**pooling**
0.15814	8	128	128	2	'sum'
0.14040	8	128	128	2	'max'
0.14330	8	128	128	3	'sum'
0.11957	8	128	128	3	'max'
0.11953	8	128	128	4	'sum'
0.11691	8	128	128	4	'max'
0.14119	8	128	128	5	'sum'
0.13384	8	128	128	5	'max'
0.17666	16	128	128	2	'sum'
0.12304	16	128	128	2	'max'
0.13462	16	128	128	3	'sum'
0.11844	16	128	128	3	'max'
0.15187	16	128	128	4	'sum'
0.11465	16	128	128	4	'max'
0.16335	16	128	128	5	'sum'
0.12103	16	128	128	5	'max'
0.19285	64	128	128	4	'sum'
0.17560	64	128	128	4	'max'
0.21796	128	128	128	4	'sum'
0.24873	128	128	128	4	'max'
⋮	⋮	⋮	⋮	⋮	⋮

4.2. táblázat. Egy pár példa a 3DGCN AE optimalizálására

MAE	**batch**	**units_conv**	**units_dense**	**num_layers**	**pooling**
0.11175	8	128	128	2	'sum'
0.12806	8	128	128	2	'max'
0.17359	8	128	128	4	'sum'
0.14074	8	128	128	4	'max'
0.14729	16	128	128	2	'sum'
0.11776	16	128	128	2	'max'
0.15849	16	128	128	4	'sum'
0.14146	16	128	128	4	'max'
0.16769	32	128	128	2	'sum'
0.11257	32	128	128	2	'max'
0.19759	32	128	128	4	'sum'
0.15193	32	128	128	4	'max'
0.19649	8	64	64	2	'sum'
0.12218	8	64	64	2	'max'
0.11712	8	64	64	4	'sum'
0.21787	8	64	64	4	'max'
0.19510	32	64	64	2	'sum'
0.14106	32	64	64	2	'max'
0.21904	32	64	64	4	'sum'
0.19487	32	64	64	4	'max'
⋮	⋮	⋮	⋮	⋮	⋮

A táblázatokban csak egy pár példát mutatok szemléltetés gyanánt, látható, hogy fontos milyen típusú poolingot használunk, vagy éppen, hogy mennyi a rétegszám.

5. fejezet

Eredmények

Mindig fontos kiértékelni a modellt, tanulmányozni az eredményeket, mert ez alapján lehet kiértékelni a modell jóságát. Ha jó eredményt kapunk, akkor jól végeztük el az adtok inicializálását, válogatását és a megfelelő hiperparaméter beállításokat választottuk. Viszont ha nem az elvárt eredményeket kapjuk, akkor folytatni kell a hiperparaméterek optimalizálását.

5.1. 3DGCN

5.1.1. A teljes TTK adathalmazon

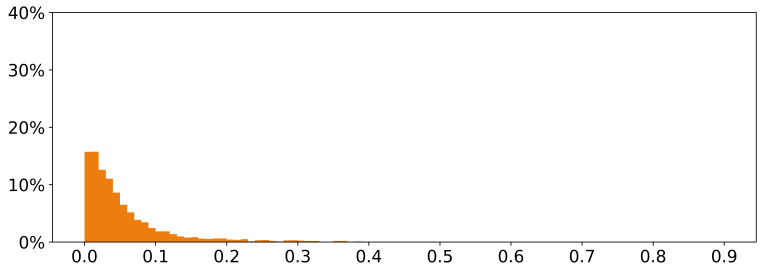
Az adathalmazt 80-10-10%-ban osztottam fel tanítási, validációs és tesztelési halmazra. A hiperparaméterek optimalizálását követve, a következő paraméterekkel végeztem el a tanítást: A *batch* méretét 16-ra, a *units_conv* és a *units_dense* 128-ra, a *num_layers* 4-re és *pooling*-nak a max poolingot használtam. 5 különböző tanítást végeztem el a tanítási, validációs és a tesztelési halmaz nagyságának változtatása nélkül, de a benne szereplő molekulákat véletlenszerűen újrarendeztem, mielőtt szét lettek volna osztva a három halmazra, tehát a *fold* értékét 5-re állítottam, az *epoch*-ot pedig 120-ra. Használok EarlyStopping-ot és ReduceLROnPlateau-t is a túltanulás elkerülése és a jobb eredmény végett. A *loss* paraméternek mse-t adtam a monitorozásnak (*monitor*) pedig a val_rmse-t. A features paramétereket pedig mindet igazra állítottam.

5.1. táblázat. A 3DGCN 5 különböző tanítás eredményei.

train MAE	**valid MAE**	**test MAE**	**train RMSE**	**valid RMSE**	**test RMSE**
0.03566	0.05673	0.05347	0.04905	0.08601	0.07956
0.02886	0.05823	0.05360	0.03961	0.08418	0.08403
0.04235	0.06448	0.06082	0.05705	0.09295	0.08954
0.03449	0.05624	0.06200	0.04745	0.07969	0.09334
0.04378	0.05560	0.06038	0.06060	0.07905	0.08812

5.2. táblázat. A 3DGCN 5 különböző tanítás eredményeinek a szórása és várható értéke.

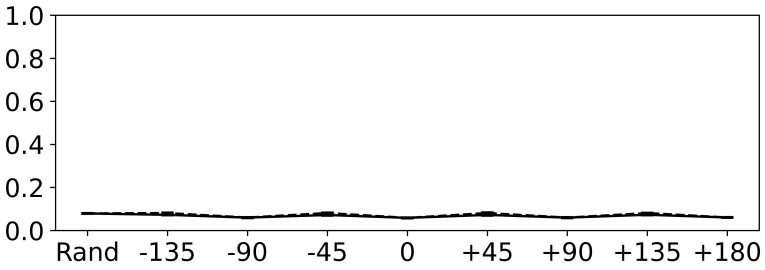
train MAE	**valid MAE**	**test MAE**	**train RMSE**	**valid RMSE**	**test RMSE**
0.03703	0.05826	0.05805	0.05075	0.08438	0.08692
0.00545	0.00323	0.00372	0.00741	0.00503	0.00473



5.1. ábra. A 3DGCN tanítás hisztogramja.

Ahogy látható a (5.2) táblázatból viszonylag alacsony a MAE[1] várható értéke és a szórása is. Várhatóan elég jó pontossággal tudja a modell elvégezni a regressziót egy molekulára, és mivel alacsony a szórása, ezért az öt különböző tanításnál nem volt fontos, hogy melyik molekula kerül a tanítási, validációs és a tesztelési halmazba.A MAE kisebb volt 0.3-nél a molekulák 98.084%-nál, kisebb volt 0.2-nél a molekulák 95.087%-nál és kisebb volt 0.1-nél a molekulák 85.087%-nál.

Ha a molekulákat elforgatjuk valamilyen szempont szerint a térben x, y, z koordinátái mentén véletlenszerűen vagy akár konkrét értékkel, akkor adatszempontjából egy új molekulát kapunk, de valójában az még attól ugyan az a molekula, nem volna szabad változnia a redukciós potenciál értékének, ami jól látható a (5.2 ábrán).



5.2. ábra. A 3DGCN forgatás invarianciája.

5.3. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.07861	0.11089
0.00305	0.00413

5.4. táblázat. 90°-kal az x tenger körül.

test MAE	test RMSE
0.05922	0.08829
0.00272	0.00299

Látható, hogy nem sokat romlott az elforgatott molekulák várható értéke és szórása, ha az eredeti állapotban lévőkön volt a tanítás elvégezve.

5.1.2. Nem a teljes TTK adathalmazon

Nem az egész (2.2.1) adathalmazon végeztem el a tanítást, hanem csak a pyr1 szetten (2580 molekula) és a nem egész pyr2 szetten (209 molekula).

[1] Mean absolute error

5.5. táblázat. A 3DGCN 5 különböző tanítás eredményeinek a szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04944	0.06277	0.06360	0.06527	0.08618	0.08769
0.02173	0.01818	0.01710	0.02814	0.02236	0.02194

5.6. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.06868	0.09307
0.01522	0.01919

5.7. táblázat. 90°-kal az x tenger körül.

test MAE	test RMSE
0.06665	0.09061
0.01729	0.02179

Itt sem romlott sokat az elforgatott molekulák várható értéke és szórása, ha az eredeti állapotban lévőkön volt a tanítás elvégezve.

5.2. 3DGCN AE

5.2.1. A teljes TTK adathalmazon

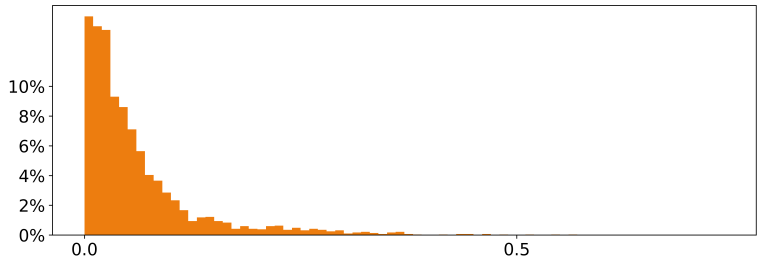
Az adathalmazt 80-10-10%-ban osztottam fel tanítási, validációs és tesztelési halmazra. A hiperparaméterek optimalizálását követve, a következő paraméterekkel végeztem el a tanítást: A *batch* méretét 8-ra, a *units_conv* és a *units_dense* 128-ra, a *num_layers* 2-re és *pooling*-nak a sum poolingot használtam. 5 különböző tanítást végeztem el a tanítási, validációs és a tesztelési halmaz nagyságának változtatása nélkül, de a benne szereplő molekulákat véletlenszerűen újrarendeztem, mielőtt szét lettek volna osztva a három halmazra, tehát a *fold* értékét 5-re állítottam, az *epoch*-ot pedig 120-ra. Használok EarlyStopping-ot és ReduceLROnPlateau-t is a túltanulás elkerülése és a jobb eredmény végett. A *loss* paraméternek mse-t adtam a monitorozásnak (*monitor*) pedig a val_rmse-t. A features paramétereket pedig mindet igazra állítottam.

5.8. táblázat. A 3DGCN AE 5 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04344	0.05942	0.06081	0.05638	0.08156	0.08546
0.04021	0.05648	0.06114	0.05376	0.07848	0.08323
0.03742	0.06166	0.05709	0.04888	0.08756	0.08080
0.03048	0.05211	0.06054	0.03952	0.07010	0.08705
0.05707	0.05803	0.06180	0.07705	0.07533	0.08451

5.9. táblázat. A (5.8) táblázat szórása és várható értéke.

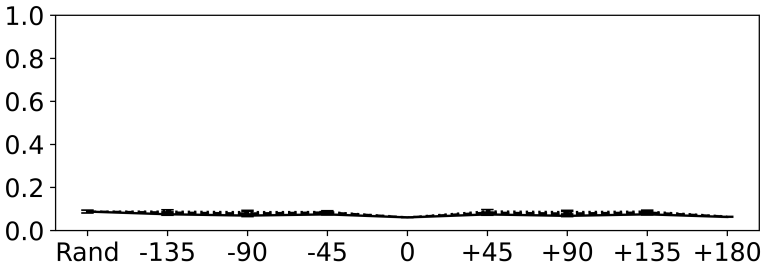
train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04172	0.05754	0.06028	0.05512	0.07861	0.08421
0.00878	0.00320	0.00164	0.01238	0.00586	0.00211



5.3. ábra. A 3DGCN AE tanítás hisztogramja.

Ahogy látható a (5.9) táblázatból viszonylag alacsony a MAE[2] várható értéke és a szórása is. Várhatóan elég jó pontossággal tudja a modell elvégezni a regressziót egy molekulára, és mivel alacsony a szórása, ezért az öt különböző tanításnál nem volt fontos, hogy melyik molekula kerül a tanítási, validációs és a tesztelési halmazba. A MAE kisebb volt 0.3-nél a molekulák 98.397%-nál, kisebb volt 0.2-nél a molekulák 94.321%-nál és kisebb volt 0.1-nél a molekulák 83.763%-nál.

Ha a molekulákat elforgatjuk valamilyen szempont szerint a térben x, y, z koordinátái mentén véletlenszerűen vagy akár konkrét értékkel, akkor adatszempontjából egy új molekulát kapunk, de valójában az még attól ugyan az a molekula, nem volna szabad változnia a redukciós potenciál értékének, ami jól látható a (5.4 ábrán).



5.4. ábra. A 3DGCN AE forgatás invarianciája.

5.10. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.04172	0.05754
0.00878	0.00320

5.11. táblázat. 90°-kal az x tenger körül.

test MAE	test RMSE
0.04172	0.05754
0.00878	0.00320

Látható, hogy nem sokat romlott az elforgatott molekulák várható értéke és szórása, ha az eredeti állapotban lévőkön volt a tanítás elvégezve.

5.2.2. Nem a teljes TTK adathalmazon

Nem az egész (2.2.1) adathalmazon végeztem el a tanítást, hanem csak a pyr1 szetten (2580 molekula) és a nem egész pyr2 szetten (209 molekula).

[2]Mean absolute error

5.12. táblázat. A 3DGCN AE 5 különböző tanítás eredményeinek a szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.02627	0.05722	0.05392	0.03339	0.07989	0.07448
0.00849	0.00675	0.00593	0.01087	0.00840	0.00884

5.13. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.06641	0.08909
0.00947	0.01307

5.14. táblázat. 90°-kal az x tenger körül.

test MAE	test RMSE
0.05592	0.07540
0.00870	0.01277

Itt sem romlott sokat az elforgatott molekulák várható értéke és szórása, ha az eredeti állapotban lévőkön volt a tanítás elvégezve.

5.3. A tanítási, validációs és a teszthalmaz nagyságának következményei

Ha csökkentjük a tanítási halmaz nagyságát, akkor fokozatosan el kezd romlani a pontosság, mivel kevesebb adaton tanul a hálózat. Minél nagyobb a tanítási halmaz annál jobb lesz az általánosító képessége, annál jobb eredményt add teljesen ismeretlen új molekulákra. Megnéztem különböző eseteket, fokozatosan csökkentve a tanítási halmazt és növelve a validációs és tesztelési halmazt.

Itt is hiperparaméterkenek ugyan azokat használtam, mint amikor a tanítási 80%, a validációs 10% és a tesztelési halmaz is 10% volt. Viszont itt nem futtattam 5 különböző tanítást, hanem csak 3-at, tehát a *fold* értékét 3-ra állítottam.

5.3.1. Tanítási 60%, validációs és tesztelési halmaz 20%

5.3.1.1. 3DGCN

5.15. táblázat. A 3DGCN 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.03372	0.05927	0.05350	0.04562	0.08579	0.07792
0.04626	0.06377	0.06511	0.06274	0.08904	0.09059
0.04296	0.06137	0.06040	0.05866	0.08992	0.08586

5.16. táblázat. A (5.15) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04098	0.06147	0.05967	0.05567	0.08825	0.08479
0.00530	0.00183	0.00476	0.00730	0.00177	0.00522

A MAE kisebb volt 0.3-nél a molekulák 98.578%-nál, kisebb volt 0.2-nél a molekulák 95.590%-nál és kisebb volt 0.1-nél a molekulák 82.942%-nál.

5.3.1.2. 3DGCN AE

5.17. táblázat. A 3DGCN AE 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04171	0.06158	0.06508	0.05461	0.08536	0.09135
0.04277	0.06219	0.06508	0.05666	0.08627	0.09099
0.02074	0.05986	0.05687	0.02638	0.08429	0.08120

5.18. táblázat. A (5.17) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.03507	0.06121	0.06234	0.04588	0.08531	0.08785
0.01014	0.00098	0.00386	0.01381	0.00081	0.00470

A MAE kisebb volt 0.3-nél a molekulák 98.056%-nál, kisebb volt 0.2-nél a molekulák 94.546%-nál és kisebb volt 0.1-nél a molekulák 82.652%-nál.

5.3.2. Tanítási 40%, validációs és tesztelési halmaz 30%

5.3.2.1. 3DGCN

5.19. táblázat. A 3DGCN 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.02509	0.06159	0.06410	0.03291	0.09105	0.09585
0.03451	0.06012	0.06445	0.04592	0.08621	0.09345
0.02469	0.07060	0.07479	0.03186	0.10147	0.10782

5.20. táblázat. A (5.19) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.02810	0.06410	0.06778	0.03690	0.09291	0.09904
0.00556	0.00567	0.00607	0.00783	0.00779	0.00770

A MAE kisebb volt 0.3-nél a molekulák 97.737%-nál, kisebb volt 0.2-nél a molekulák 94.138%-nál és kisebb volt 0.1-nél a molekulák 82.298%-nál.

5.3.2.2. 3DGCN AE

5.21. táblázat. A 3DGCN AE 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.05591	0.07415	0.07583	0.07298	0.10246	0.10382
0.04029	0.06841	0.07212	0.05176	0.09239	0.09917
0.02403	0.07612	0.07115	0.03058	0.10708	0.09859

5.22. táblázat. A (5.21) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04007	0.07289	0.07303	0.05178	0.10064	0.10053
0.01301	0.00327	0.00201	0.01731	0.00613	0.00233

A MAE kisebb volt 0.3-nél a molekulák 97.845%-nál, kisebb volt 0.2-nél a molekulák 92.455%-nál és kisebb volt 0.1-nél a molekulák 77.887%-nál.

5.3.3. Tanítási 20%, validációs és tesztelési halmaz 40%

5.3.3.1. 3DGCN

5.23. táblázat. A 3DGCN 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.02468	0.07060	0.07478	0.03185	0.10146	0.10782
0.03120	0.07504	0.07713	0.04048	0.10768	0.11002
0.03228	0.08195	0.07910	0.04138	0.11787	0.11253

5.24. táblázat. A (5.23) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.02939	0.07586	0.07700	0.03790	0.10900	0.11012
0.00335	0.00467	0.00176	0.00429	0.00676	0.00192

A MAE kisebb volt 0.3-nél a molekulák 96.939%-nál, kisebb volt 0.2-nél a molekulák 91.456%-nál és kisebb volt 0.1-nél a molekulák 76.168%-nál.

5.3.3.2. 3DGCN AE

5.25. táblázat. A 3DGCN AE 3 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.03909	0.08481	0.08814	0.05127	0.11555	0.11886
0.05402	0.08463	0.08282	0.06963	0.11369	0.11132
0.03171	0.08010	0.08058	0.04043	0.10891	0.10891

5.26. táblázat. A (5.25) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.04160	0.08318	0.08385	0.05378	0.11272	0.11303
0.00927	0.00218	0.00316	0.01205	0.00279	0.00423

A MAE kisebb volt 0.3-nél a molekulák 96.258%-nál, kisebb volt 0.2-nél a molekulák 90.499%-nál és kisebb volt 0.1-nél a molekulák 72.005%-nál.

5.4. QM7b

A QM7b adathalmaz (2.2.2) 14 tulajdonságot tartalmaz, ez közül az egyik az ionizációs potenciál. Elvégeztem a tanítást a 3DGCN és a 3DGCN AE modellekkel az ionizációs potenciálra. A tanítás előtt hiperparaméter optimalizálást végeztem, így a megfelelő paraméterekkel tudtam tanítani. A *fold* értékét 5-re állítottam, 5 különböző tanítást végeztem el, a következő eredményeket kaptam.

5.4.1. 3DGCN

5.27. táblázat. A 3DGCN 5 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.06907	0.13398	0.13497	0.09046	0.20134	0.19801
0.06168	0.13155	0.12893	0.07859	0.19265	0.18279
0.04305	0.12324	0.12507	0.05499	0.17639	0.18731
0.08301	0.12374	0.12446	0.10780	0.16876	0.16975
0.07312	0.12106	0.12594	0.09486	0.16439	0.18424

5.28. táblázat. A (5.27) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.06599	0.12671	0.12787	0.08534	0.18071	0.18442
0.01337	0.00508	0.00386	0.01782	0.01411	0.00906

A MAE kisebb volt 0.3-nél a molekulák 91.761%-nál, kisebb volt 0.2-nél a molekulák 82.386%-nál és kisebb volt 0.1-nél a molekulák 55.506%-nál.

5.4.2. 3DGCN AE

5.29. táblázat. A 3DGCN AE 5 különböző tanítás eredményei.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.05379	0.13099	0.12661	0.06663	0.18139	0.17129
0.06029	0.12267	0.13037	0.07793	0.16651	0.17398
0.10053	0.14251	0.13909	0.12729	0.18633	0.18136
0.05756	0.12654	0.12973	0.07262	0.17466	0.17851
0.05319	0.12312	0.13456	0.06585	0.16828	0.19229

5.30. táblázat. A (5.29) táblázat szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.06507	0.12917	0.13207	0.08206	0.17543	0.17948
0.01791	0.00730	0.00432	0.02303	0.00755	0.00729

A MAE kisebb volt 0.3-nél a molekulák 91.096%-nál, kisebb volt 0.2-nél a molekulák 81.026%-nál és kisebb volt 0.1-nél a molekulák 55.146%-nál.

5.5. FreeSolv

A FreeSolv adathalmaz a molekuláknak az oldhatóságát tartalmazza, az is egy nagyon fontos tulajdonság. Itt is elvégeztem a tanítást a 3DGCN és a 3DGCN AE modellekkel az oldhatóságra.

A 3DGCN AE valamivel jobb eredményt adott, mint a 3DGCN.

5.5.1. 3DGCN

5.31. táblázat. A 3DGCN különböző tanítás szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.26848	0.57491	0.66469	0.33950	0.80132	0.948832
0.06198	0.08071	0.06197	0.07677	0.13435	0.115354

5.32. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.69388	0.98767
0.07307	0.11970

5.33. táblázat. 45°-kal az x tenger körül.

test MAE	test RMSE
0.64604	0.91382
0.06648	0.12298

5.5.2. 3DGCN AE

5.34. táblázat. A 3DGCN AE különböző tanítás szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.27524	0.54820	0.64025	0.35359	0.75428	0.91908
0.05731	0.10912	0.08733	0.06880	0.16627	0.14777

5.35. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.65953	0.90128
0.09672	0.15833

5.36. táblázat. 45°-kal az x tenger körül.

test MAE	test RMSE
0.59756	0.83214
0.09183	0.14331

5.6. ESOL

Az ESOL adathalmaz a molekuláknak az oldhatóságát tartalmazza, az is egy nagyon fontos tulajdonság. Elvégeztem a tanítást a 3DGCN és a 3DGCN AE modellekkel az oldhatóságra.

A 3DGCN AE valamivel jobb eredményt adott itt is, mint a 3DGCN.

5.6.1. 3DGCN

5.37. táblázat. A 3DGCN különböző tanítás szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.22225	0.46694	0.49706	0.27621	0.59038	0.63317
0.03150	0.03469	0.03059	0.03870	0.03168	0.04466

5.38. táblázat. Véletlenszerű elforgatás.

test MAE	test RMSE
0.51460	0.65107
0.03328	0.04350

5.39. táblázat. 90°-kal az x tenger körül.

test MAE	test RMSE
0.51499	0.65049
0.03854	0.06050

5.6.2. 3DGCN AE

5.40. táblázat. A 3DGCN AE különböző tanítás szórása és várható értéke.

train MAE	valid MAE	test MAE	train RMSE	valid RMSE	test RMSE
0.23067	0.43820	0.47541	0.28648	0.55293	0.61400
0.05430	0.03360	0.03846	0.06691	0.04257	0.05143

5.41. táblázat. Véletlenszerű elforgatás.

test MAE	**test RMSE**
0.49279	0.62176
0.04688	0.05433

5.42. táblázat. 90°-kal az x tenger körül.

test MAE	**test RMSE**
0.49676	0.62878
0.04923	0.06497

6. fejezet

Összefoglalás

Dolgozatom célja olyan eljárások és megoldási lehetőségek vizsgálata, kipróbálása, tovább fejlesztése, amelyek neurális háló segítségével regressziót végeznek a molekulák egyes tulajdonságaira. Ehhez a szakirodalom kutatás során korábban megalkotott modelleket elemeztem, vizsgáltam az egyes modelleknek az előnyeit és hátrányait. Azt tapasztaltam, hogy ha a molekulákat gráfként ábrázoljuk, akkor viszonylag elég jó eredményeket lehet elérni. Jó kombináció a konvolúció és a gráfok ötvözése. Fontos az adatok megfelelő előkészítése, ha szükséges akkor az augmentálása vagy éppen valamilyen konverzió elvégzése. Nem csak az adathalmaz jósága, hanem a megfelelő optimalizálások elvégzése is szükséges, hiperparaméter optimalizálás nélkül rosszabb eredményt kapnánk.

A 3DGCN az egy nagyon jó modell, jó alapkoncepcióra épül, de ki lehet azért bővíteni. Hasznosnak ítélem az adatok augmentálhatóságának a kibővítését, valamint a sokkal jobban szabályozható adathalmaz felosztást, ami nem csak a hiperparaméterek optimalizálásnál nyújt előnyt. Megpróbáltam növelni a 3DGCN általánosítóképességét azzal, hogy autoenkóder jellegű hálózattá alakítottam. Amikor nem az egész adathalmazon végeztem el a tanítást, validációt és tesztelést, akkor jobbnak bizonyult az autoenkóderes modell, jobb volt az általánosítóképessége. Ugyan úgy az ESOL és a FreeSolv adathalmazoknál is jobban teljesített az autoenkóderes változat, általánosabbak a molekulák és kisebb volt az adathalmaz. Ha az adathalmazon belül csökkentjük a tanítási halmaz nagyságát és ezzel szemben folyamatosan növeljük a validációs és teszthalmazt, akkor egyre rosszabb eredményt fogunk kapni.

6.1. Továbbfejlesztési lehetőségek

Ki lehetne próbálni az autoenkódernek további fajtáit is, mint például a variációs autoenkódert. A skaláris és a vektoriális jellemzőkön is lehetne még finomítani. Abban is lenne potenciál, ha áttérnénk polárkoordinátákra, mivel az atomok x, y, z koordinátaiból ki lehetne számolni, hogy milyen távol vannak egymástól az atomok és a bezárt szögeket. A modell egyszerűsítése lehetne az, ha áttérnénk két dimenzióba, a térbeli molekulákat, gráfokat kisimítanánk és csak síkokban dolgoznánk. Számtalan módon lehetne folyatni, minél jobban beleássa magát az ember, annál több lehetőség tárul fel.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani felelős konzulensemnek Dr. Szegletes Lucának, aki bevezetett a gráfkonvolúciók világába, segítséget nyújtott a téma átlátásában, az elméleti háttér elsajátításában, továbbá nagyon hálás vagyok a tanácsaiért, szüntelen ösztönzéséért.

Köszönettel tartozok Hamza Andreának a Természettudományi Kutatóközpont tudományos főmunkatársának, aki segített megérteni a molekulák kémiai hátterét.

A dolgozatban bemutatott kutatásokat az Európai Unió H2020 programjával támogatta (875565-CompBat projekt, LC-BAT-3-2019).

Irodalomjegyzék

[1] D. Weininger. http://organica1.org/seminario/smile1988.pdf. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules, J. Chem. Inf. Comput. Sci. 1988, 28, pp. 31-36. (2020.10.27.).

[2] Jürgen Bajorath. http://infochim.u-strasbg.fr/CS3/program/material/Bajorath.pdf. Department of Life Science Informatics, LIMES Program Unit Chemical Biology University of Bonn (2020.10.27.).

[3] http://quantum-machine.org/datasets/. QM7b Dataset (2020.10.27.).

[4] L. C. Blum and J.-L. Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *J. Am. Chem. Soc.*, 131:8732, 2009.

[5] Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *New Journal of Physics*, 15(9):095003, 2013. URL http://stacks.iop.org/1367-2630/15/i=9/a=095003.

[6] Evan N. Feinberg Joseph Gomes Caleb Geniesse Aneesh S. Pappu Karl Leswing Vijay Pande Zhenqin Wu, Bharath Ramsundar. http://moleculenet.ai/; https://arxiv.org/pdf/1703.00564.pdf. MoleculeNet: A Benchmark for Molecular Machine Learning, arXiv preprint, arXiv: 1703.00564, 2017 (2020.10.27.).

[7] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5028207/#FD1. Molecular graph convolutions: moving beyond fingerprints (2020.10.27.).

[8] https://towardsdatascience.com/how-to-do-deep-learning-on-graphs-with-graph-convolutional-networks-7d2250723780, . Towards Data Science (2020.10.27.).

[9] https://towardsdatascience.com/how-to-do-deep-learning-on-graphs-with-graph-convolutional-networks-62acf5b143d0/, . Towards Data Science (2020.10.27.).

[10] Hyeoncheol Cho and Insung Choi. Three-dimensionally embedded graph convolutional network (3dgcn) for molecule interpretation. *arXiv preprint arXiv:1811.09794*, 2018.

[11] Will Badr. https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726. Auto-Encoder (2020.10.27.).

[12] Stuart Russell, Peter Norvig, Róbert Édelkraut, Péter Antal, Tadeusz Dobrowiecki, Géza Haidegger, Tamás Haidegger, Gergely Héja, Tamás Mészáros, Béla Pataki, Gyula Román, György Strausz, Károly Tilly, József Valyon, Péter Varga, and Annamária Várkonyiné Kóczy. *Mesterséges Intelligencia (Modern megközelítésben)*. Panem Könyvkiadó, 2005. http://mialmanach.mit.bme.hu/aima/ch01s01/.

[13] Altrichter Márta, Horváth Gábor, Pataki Béla, Strausz György, Takács Gábor, and Valyon József. *Neurális hálózatok*. Panem Könyvkiadó, 2006. http://mialmanach.mit.bme.hu/neuralis/ch01s01/.

[14] https://www.rdkit.org/docs/index.html. RDKit (2020.10.27.).

[15] https://github.com/TobiasSkovgaardJepsen/posts/blob/master/HowToDoDeepLearningOnGraphsWithGraphConvolutionalNetworks/Part2_SemiSupervisedLearningWithSpectralGraphConvolutions/notebook.ipynb. (2020.10.27.).

[16] https://stellargraph.readthedocs.io/en/stable/demos/node-classification/gcn-node-classification.html, . StellarGraph (2020.10.27.).

[17] https://stellargraph.readthedocs.io/en/stable/demos/link-prediction/gcn-link-prediction.html, . StellarGraph (2020.10.27.).

[18] https://missinglink.ai/guides/convolutional-neural-networks/graph-convolutional-networks/, . missinglink (2020.10.27.).

[19] Y. Bengio Y. LeCun, L. Bottou and P. Haffner. http://yann.lecun.com/exdb/mnist/index.html, 1998. "Gradient-based learning applied to document recognition" Proceedings of the IEEE, 86(11):2278-2324 (2020.10.27.).

[20] Arden Dertat. https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798. Auto-Encoder (2020.10.27.).

[21] http://www.kislexikon.hu/oxidaloszer.html. KISLEXIKON (2020.10.27.).

[22] http://www.kislexikon.hu/redukaloszer.html, . KISLEXIKON (2020.10.27.).

[23] https://labornite.hu/2019/06/15/redoxpotencial-orp-merese/, . labornite (2020.10.27.).

[24] Hamza Andrea, a TTK, azaz a Természettudományi Kutatóközpont munkatársa.

[25] Experimental and Calculated Electrochemical Potentials of Com- mon Organic Molecules for Applications to Single-Electron Redox Chemistry Hudson G. Roth, Nathan A. Romero and David A. Nicewicz, Synlett 2016, 27, A–J.

[26] https://megnyitasa.com/extension/xyz. (2020.10.27.).

[27] https://en.wikipedia.org/wiki/Chemical_table_file. KISLEXIKON (2020.10.27.).

[28] https://openbabel.org/docs/dev/Installation/install.html. Open Babel (2020.10.27.).