Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Aplikace metod pro rozšíření datové sady EEG záznamu

Plzeň 2022 Jaroslav Lehečka

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jaroslav LEHEČKA**
Osobní číslo:	**A19B0121P**
Studijní program:	**B0613A140015 Informatika a výpočetní technika**
Specializace:	**Informatika**
Téma práce:	**Aplikace metod pro rozšíření datové sady EEG záznamu**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Prostudujte problematiku zpracování EEG signálu, evokovaných potenciálů a seznamte se s dostupnými metodami pro rozšiřování datové sady.
2. Na základě bodu 1 vyberte metody a navrhněte způsob rozšíření záznamů v existující datové sadě evokovaných potenciálů.
3. Navržené řešení implementujte v jazyce Python.
4. Využijte původní a rozšířenou datovou sadu v klasifikační úloze.
5. Porovnejte výstupní metriky klasifikační úlohy pro původní a rozšířený dataset.

Rozsah bakalářské práce: **doporuč. 30 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

Vedoucí bakalářské práce: **Ing. Pavel Šnejdar**
Katedra informatiky a výpočetní techniky

Datum zadání bakalářské práce: **4. října 2021**
Termín odevzdání bakalářské práce: **5. května 2022**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 14. října 2021

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 23. června 2022

Jaroslav Lehečka

Abstract

These days is research on dementia is connected with computer science. Data processing could be performed using various methods - mainly machine learning. These methods require sufficient input data in the appropriate quality. This is the only way to reach the set valules. The aim of this bachelor thesis is generating synthetic datasets for larger ones and more robust algorithms using studied methods. Different methods were introduced and applied (shift, noise, combination, GAN network). With changes made on input datasets, would be a classifier prepared for noisy datasets or worse measurement conditions. The results confirm that the usage of machine learning is really possible for dataset augmentation.

Abstrakt

V současnosti je výzkum demence spojen s počítačovou vědou. Zpracovávání dat může být provedeno pomocí různých metod - hlavně strojového učení. Tyto metody vyžadují dostatek vstupních dat v náležité kvalitě. To je jediná cesta jak dosáhnou stanovených hodnot. Cílem této bakalářské práce je dogenerování syntetických dat pro větší robustnost klasifikačních algoritmů. Využito bylo různých metod (posun, šum, kombinace, GAN síť). Úpravou vstupních dat může být klasifikátor připraven i pro situace, kde je významně větší rušení nebo horší podmínky měření. Výsledky potvrzují, že použití strojového učení je možné pro rozšiřování datových souborů.

Obsah

1 Úvod

Informatika je moderní vědní obor, který zasahuje do mnoha oblastí lidské činnosti. Zajímavé uplatnění najdeme také v oblasti neurovědy.

Činnost mozku lze sledovat prostřednictvím elektroencefalografu (EEG). Data, která získáme, můžeme zpracovat informačními technologiemi. Výsledky toho měření mohou přispět například i k výzkumu demence. S touto nemocí měla naše rodina osobní zkušenost, a proto velice oceňuji každý pokrok v této oblasti.

Při zpracování dat z EEG lze s úspěchem použít metod strojového učení. Podmínkou pro využití této metody je dostatek vstupních dat ve vhodné kvalitě [32].

Získat dostatek dat je velmi obtížné (délka měření, šum, množství lidí), proto byly vynalezeny techniky pro rozšiřování datových souborů. Klasifikátory jsou následně kvalitnější a robustnější. Jsou připraveny i na jiné datové soubory, než se kterými byli naučeny.

Předmětem této bakalářské práce je příprava dat pro rozšíření datové sady. Budou navrženy techniky, které zajistí rozptyl ve vstupních datech. Následně bude řešení otestováno na původní i rozšířené datové sadě na základě výstupních metrik klasifikační úlohy. Pomocí vybraných algoritmů bude pak navržen způsob rozšíření záznamů v existující datové sadě evokovaných potenciálů. Výsledkem práce budou nové datové vzorky, které umožní natrénování více robustního klasifikačního algoritmu.

2 Mozková aktivita, její zobrazení a zpracování získaných dat

Mozek je nejdůležitější částí lidského těla. Řídí nejen činnost všech dalších orgánů, ale též reakce na podněty a vědomé jednání.

Základní stavební jednotkou nervové soustavy je nervová buňka - neuron. Lidský mozek jich obsahuje miliardy. Ke každému neuronu patří 20-1000 synapsí (spojení s jinými neurony).

Neurony dokáží rychle přenášet informace ve formě podráždění. To je umožněno specializovanou cytoplasmatickou membránou neuronu. Vzruch je přenášen díky sodíko-draselným (NaK) pumpám.

Vnitřní povrch membrány nese záporný náboj, vnější kladný náboj. Vzniká pak membránový potenciál, který činí -50 až -90 mV [28]. Při vzruchu dojde k depolarizaci membrány a rychlému poklesu membránového potenciálu.

Zachytit činnost nervové soustavy je možné, pokud dokážeme zaznamenat změny elektrického napětí při šíření vzruchu. [30]

2.1 Měření EEG signálů

Elektroencefalografie je diagnostická metoda záznamu činnosti nervové soustavy. Elektroencefalograf (EEG) zachycuje změny v polarizaci neutronů. Porovnává elektrický potenciál dvou bodů na kůži lebky (bipolární záznam) nebo bod aktivní mozkové tkáně s neaktivním bodem umístěným například na ušním boltci (unipolární záznam). [17]

Při zkoumání činnosti mozku sledujeme především **evokované potenciály**. Ty představují reakci nervové soustavy na nějaký smyslový podnět, který se projeví významnou změnou v EEG signálu. Sledovat můžeme rychlost reakce, zapojení jednotlivých částí mozku, tvar, polaritu a vzájemné vztahy jednotlivých vln. Vyšetřování evokovaných potenciálů představuje funkční zhodnocení určité senzorické dráhy. [26]

2.1.1 Metody snímání

Metody snímání můžeme rozlišovat podle přímého zásahu do šedé kůry mozkové na invazivní a neinvazivní.

Invazivní metody

Během operací mozku se využívá snímání přímo z šedé kůry mozkové. Přesněji se takové měření označuje elektrokortikografie. Výhodou je měření přímo na tkáních, případně hlouběji v mozku. Při měření nepřekáží kůže, vlasy ani lebka. Výsledky jsou tedy přesnější. Tato metoda se také používá při zaměřování ložisek různých nemocí (například epilepsie) v rámci mozkové kůry. [5]

Neinvazivní metody

Alternativou k invazivním metodám jsou metody neinvazivní. Své uplatnění najdou mezi rutinními neurologickými vyšetřeními nebo u vědeckých výzkumných projektů. Tento druh vyšetření je pro pacienta nezatěžující a bezbolestný.

Při měření z povrchu hlavy je umístění elektrod přesně určeno. Při větším počtu záznamových kanálů se využívá čepice, která slouží jako nosič elektrod a zajišťuje jejich vhodné uchycení a správné umístění. Mezi elektrodu a pokožku je nanesen vodivý gel.

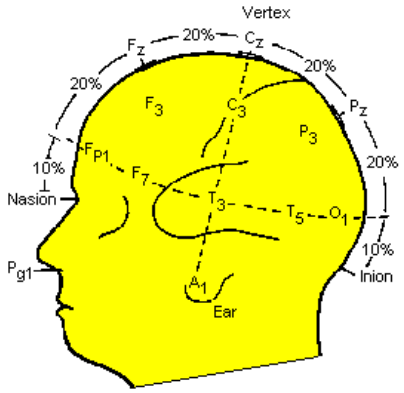
Rozložení elektrod na hlavě je mezinárodně standardizováno, jejich označení je odvozeno od anglického pojmenování pozice elektrody (viz obrázek 2.6). [5]

2.1.2 Elektrody

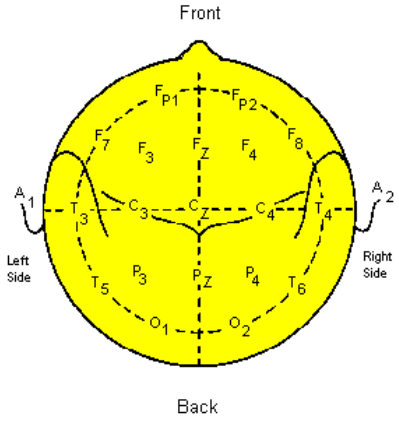
Označení elektrody se skládá z kombinace písmena a číslice. Název je odvozen od pozice elektrody a části mozku mozku, jejíž činnost snímá. Písmeno odznačuje pozici elektrody v rámci předozadní lokalizace elektrody. [15]

- Fp - frontpolární pozice
- F - frontální pozice
- P - parientální pozice
- C - centrální pozice
- O - okcipitální pozice
- T - temporální

K rozmisťování elektrod se využívá několika systémů. Mezi nejznámější patří *10 - 20 System*, který se odkazuje na 10% nebo 20% mezielektrodovou vzdálenost. Rozmístění najdete na obrázku 2.2 a 2.6.[2]



Obrázek 2.1: Rozmístění elektrod EEG - boční pohled [2]



Obrázek 2.2: Rozmístění elektrod EEG - pohled zhora [2]

Pro záznam aktivity centrální nervové soustavy se používají různé elektrody, které se liší tvarem, materiálem a způsobem užití. Mohou být jednorázové i použitelné opakovaně, integrované do čepice nebo čelních pásků. Na operačních sálech se využívají jehlové elektrody, které se dostanou až pod pokožku hlavy.

Vzhledem k nutnosti zaznamenat rychlé změny signálu je potřeba, aby byly elektrody nepolarizovatelné. To ovšem omezuje využití některých materiálů. V současné době se užívají elektrody z čistého stříbra nebo ze slitiny stříbra a cínu obalené chloridem stříbrným. V současné době jsou zkoumány metody, které by umožnily rozvoj slitin a ocelí určených pro lepší snímání obecných signálů a tlumení reakcí mezi různými materiály. Důležitá je v rámci přenosu signálu i volba gelu, který může reagovat s elektrodou a tedy ovlivnit její vlastnosti. [26]

2.1.3 Snímané vlny

Evokované potenciály nám pomáhají nahlédnout do fungování našeho mozku. Můžeme tak sledovat reakce na různé podněty působící na naše smysly. Tvoří je sled negativních a pozitivních vln, které se odlišují frekvencí, amplitudou a latencí.

Frekvence hodnotí počet opakujících se vln v určitém časovém úseku. Amplituda zachycuje velikost odezvy mozku. Latencí rozumíme čas mezi působením podnětu a reakcí mozku.

Podle latence rozeznáváme několik druhů vln. Písmeno v jejich názvu označuje polaritu (P – positive, kladná, N – negative, záporná, C – bez jednoznačně určené polarity). Číslice uvádí přibližné zpoždění (ve stovkách milisekund).

Mezi nejznámější patří vlny C1, P1, N1 a P3 (někdy je označována i P300).

- Vlna C1 je první vlnou, která se objeví (zhruba 60-95 ms) po přijetí stimulu. Reaguje na vizuální podnět a je plně závislá na tom, kde v zorném poli ke stimulu došlo. Při vícenásobném opakování stejného stimulu je vlna víceméně neměnná.

- Vlna P1 následuje vlnu C1 zhruba 80-135 ms po objevení stimulu. Vlnu P1 můžeme označit jako první pozitivní vlnu. S přibývajícím věkem se limitní hodnoty této vlny prodlužují (115 ms do 60 let, později 120/125 ms). Pomocí amplitudy této vlny můžeme poznat, zda stimul nastal v oblasti, na kterou měl testovaný subjekt soustředěnou pozornost.

- Vlna N1 se objevuje zhruba 100 - 150 ms po stimulu. Tato vlna je na rozdíl od předchozích vln negativní. Stejně jako P1 vlna je vázána na pozornost testovaného.

- Vlna P3 (P300) se objevuje po skončení zpracování stimulu. Do této doby (300- 500 ms) jsou fakticky veškeré stimuly neuvědomělé.

- Vlna N400, která se objevuje 250-500 ms po stimulu, reprezentuje stav. Je vázána na rozpoznání známého symbolu, tváře, slova nebo zvuku.

Z hlediska frekvenčního rovněž rozeznáváme několik druhů vln. Všechny jsou pozorovatelné jak u dětí, tak u dospělých.

- Delta vlny (do 4 Hz) najdeme při nejnižší činnosti mozku, tedy non-REM spánku nebo při kómatu.

- Théta vlny (4 až 7 Hz) spojujeme se stavem větší únavy, ale i s depresemi.
- Alfa vlny (8 až 12 Hz) symbolizují stav zavřených očí a relaxaci.
- Beta vlny (12 až 30 Hz) zaznamenáme při bdělém stavu, zejména při analýze a dedukci.
- Gama vlny (více než 30 Hz) se vyskytují při vysokém vypětí a v situacích velmi náročných na pozornost. Uplatňují se mimo jiné v rámci řídících, učících a paměťových procesů. [19]

2.1.4 P3 vlny

Vlna P3 (někdy P300) je evokovaný potenciál zaznamenatelný pomocí elektroencefalografu se zpožděním 300-500 ms po výskytu stimulu. Stimul může být buď zvukový nebo vizuální, případně kombinovaný. Příkladem mohou být obraz, zvuk nebo video.

V roce 1975 bylo dosaženo prvního významnějšího rozlišení vlny P3a (extrém na frontálním laloku Fz) a P3b (extrém na parentálním laloku Pz). Pro klasifikátor byly využity pouze vlny P3b.

P3 vlny jsou dobře pozorovatelné pro očekávané stimuly. Jejich význam v činnosti mozku nebyl zatím zcela objasněn. Jednou z nejpravděpodobnějších teorií je, že vlna vzniká v okamžiku, kdy je potřeba aktualizace pracovní paměti. Navzdory tomu, že nevíme s jistotou, k čemu tato vlna slouží, můžeme ji změřit a určit, kterou část mozku ovlivňuje.

2.1.5 Postup snímání - Odball experiment

Oddball experiment je design experimentu využívaného v rámci psychologického výzkumu. Stimulované osobě jsou zobrazovány stálé subjekty, které jsou v nepravidelných časových intervalech nahrazeny subjekty jinými. Tyto záměny vyvolávají u měřené osoby stimul, který je následně zkoumán.

2.2 Zpracování EEG signálu

Elektroencefalograf musí zaznamenat velmi malé změny elektrického napětí. Měření navíc může ovlivnit řada vnějších i vnitřních faktorů. Pro vědecké zkoumání je vhodné zajistit, aby bylo měření co nejméně zkresleno. Aby byly výsledky měření elektroencefalografu vůbec použitelné, je třeba jeho signál strojově upravit.

2.2.1 Problémy při zpracování EEG dat

Zaznamenaný EEG signál obsahuje kromě nosných dat také jevy, které jsou nežádoucí. Jedná se nejen o jevy technické (šum), ale také sociální (ruch v okolí ovlivňuje zkoumanou osobu). Problém také může nastat při nedostatečném množství naneseného gelu na elektrodě (horší přenos).

Pro vyhodnocení EEG dat je třeba získat velké množství záznamů. Výsledek totiž není často patrný po jednom měření, v některých případech je potřeba získat i několik stovek záznamů. Až po jejich zprůměrování lze signál relevantně vyhodnotit.

Některé jevy se těžko eliminují. Jsou to většinou jevy biologické, ke kterým můžeme zařadit například mrkání nebo kašlání pacienta při měření. Nejzastoupenější množinu tvoří svalové kontrakce. Negativní vliv může mít i stres a nervozita. Data jsou pak pro strojové učení hůře použitelná. Proto je nutné pomocí augmentačních metod získaná data rozšířit a uzpůsobit tomu případný experiment.

Jelikož získaný signál z elektrod má velice nízkou amplitudu (již z podstaty sbírání na pokožce je signál tlumen, tedy dosahuje amplitud v řádu několika mikrovolt), je nutné jej před převodem na digitální signál zesílit. Jedním z jednodušších způsobů, jak velmi slabý vstupní signál zesílit, je použití zesilovače v diferenčním zapojení. Na přímý a na referenční vstup se přivedou signály z elektrod, na výstupu najdeme napěťový rozdíl mezi oběma vstupy. Výsledkem je signál s omezeným šumem.

2.2.2 Evokované potenciály

Evokované potenciály jsou reakce nervové soustavy vyvolané nějakým stimulem. Projevují se jako změna elektrické aktivity mozku a mohou být zaznamenány elektroencefalografem. Na jednom záznamu není změna jasně patrná, a tak je třeba zachytit evokovaný potenciál opakovaně a výsledky zprůměrovat. Čím větší je evokovaná odpověď a čím menší je její rušení vnitřními a zevními vlivy, tím menší počet zprůměrněných záznamů je nutný. Někdy stačí 50, někdy i několik tisíc.

Reakce nervové soustavy je zachycena jako negativní a pozitivní vlny. U evokovaných potenciálů sledujeme především rychlost vedení, se kterou se vzruchy šíří v jednotlivých drahách. Ta se projeví jako latence reakce. Další sledované parametry jsou amplituda a tvar odpovědi. Tyto veličiny závisí mimo jiné na množství činných neuronů a na stupni nebo stadiu synchronizace, se kterou se vzruchy šíří nervovými drahami.

Evokované potenciály můžeme rozdělit do dvou skupin:

- Endogenní potenciály (jsou nezávislé na vnějších receptorech, ale jsou ovlivněny momentálním psychickým stavem subjektu)
- Exogenní potenciály (závisí na stimulované smyslové oblasti, a jsou proto vhodnější pro pozorování)

2.2.3 Druhy evokovaných potenciálů

Mezi základní druhy evokovaných potenciálů řadíme sluchové, zrakové nebo somatosenzorické evokované potenciály.

Zrakově evokované potenciály

Charakter podnětu má vliv na to, která část zrakového systému bude podrážděna a jaká bude senzitivita použitého testu. Jako vizuální podněty se často využívají stimulace pomocí záblesků nebo strukturovaný podnět (černobílá šachovnice s pravidelnou změnou barev). Nejvýznamnějším parametrem pro hodnocení těchto evokovaných potenciálů je latence a amplituda P100.

Sluchové evokované potenciály

Jsou to reakce nervové soustavy na sluchový podnět. Během vyšetření je druhé ucho blokováno tzv. „bílým šumem“, aby se zabránilo stimulaci kostním vedením. Normální zvukový evokovaný potenciál je tvořen sérií pozitivních vln s latencí do 10 ms. Lze však registrovat i odpovědi s latencí střední (10-50 ms) a pomalou (50-300 ms). Intenzita stimulu ovlivňuje latenci i amplitudu.

Somatosenzorické evokované potenciály

Reakce nervové soustavy na podráždění periferních nervů. Odpověď může být vyvolána elektrickými stimuly, laserem nebo krátkým proudem vzduchu. Motorické evokované potenciály jsou na rozdíl od senzitivních potenciálů snadno registrovatelné a nevyžadují průměrování.

2.2.4 Formát dat

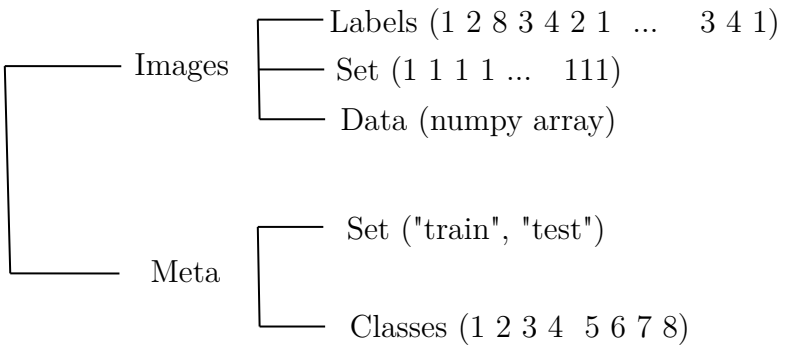
Získávání a sběr dat je velmi náročná činnost. Při snímání EEG signálu se předpokládá, že naměřený signál bude použit při mnoha experimentech. Důležité při měření jsou nejen podmínky, za jakých byl experiment prováděn,

ale také datový formát, který byl použit při záznamu. V současné době můžeme získat různá data v mnoha rozdílných datových formátech. Při ukládání je důležité, aby byly dodrženy základní principy pro uchovávání dat.

Při uchovávání dat se předpokládá dodržení tzv. FAIR principů.

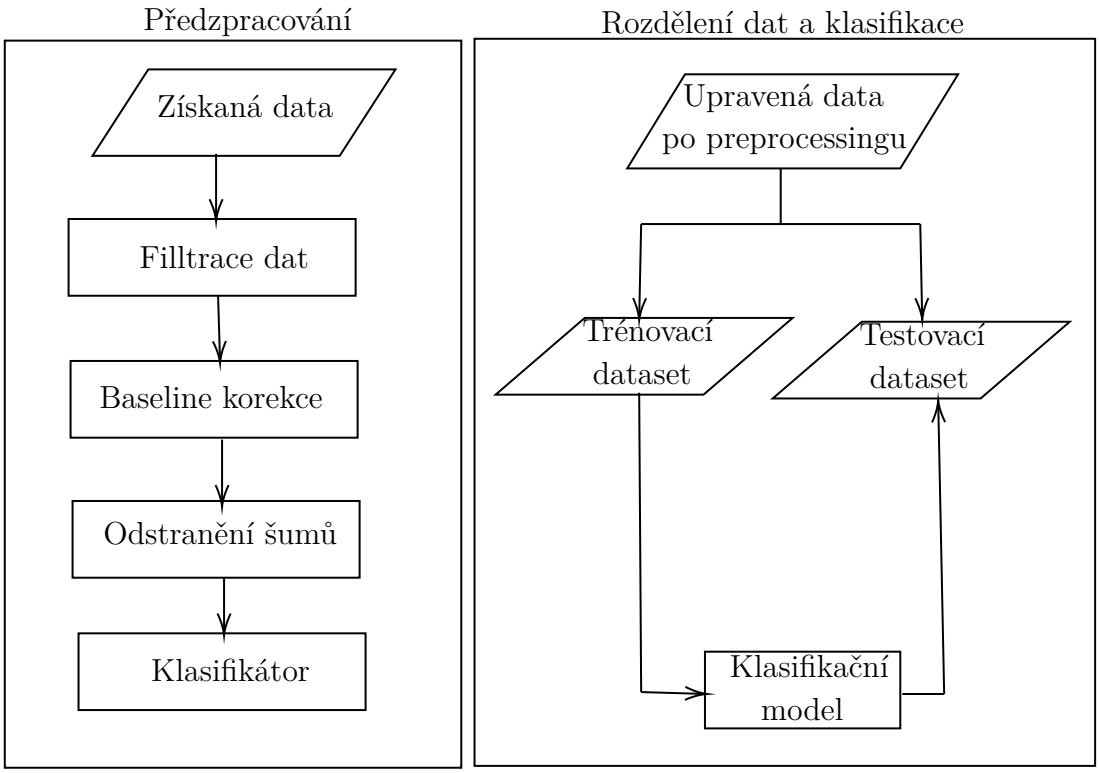
- Dohledatelnost
  - (Meta)data mají přiřazen globálně jedinečný stálý identifikátor
  - Veškerá (meta)data jsou detailně popsána (meta)daty
  - Indexy jsou prohledávatelné
- Přístupnost
  - (Meta)data jsou dosažitelná na základě identifikátorů za pomoci standardizovaných protokolů
  - Protokol je volně dostupný a univerzálně implementovatelný
  - Je možné zabezpečení vůči cizímu přístupu
- Interoperabilita
  - (Meta)data používají formální, přístupný, hromadně rozšířený jazyk pro reprezentaci znalostí
  - (Meta)data používají slovní zásobu řízenou pravidly FAIR
  - (Meta)data obsahují kvalifikované odkazy na jiná (meta)data
- Znovupoužitelnost
  - (Meta)data používají množství přesných a relevantních atributů
  - (Meta)data jsou vydávána s jasnou licencí k použití
  - (Meta)data jsou spojena se svým původem
  - (Meta)data splňují standardy z dané oblasti

V současné době probíhá výzkum ohledně standardizace dat a formátu. Předmětem projektu NIX (Pandora) je standardizace metod a modelů pro ukládání EEG a ostatních neurovědeckých dat ve formátu založeném na HDF5. V rámci projektu byly vytvořeny i užitečné knihovny v jazyce C++ pro usnadnění vývoje a práce s daty.



Obrázek 2.3: Struktura .MAT souboru

Jednou z knihoven splňující FAIR principy je NEO. Pro naprogramování bylo využito jazyka Python. Hlavní výhodou je možnost převodu dat od různých výrobců, dokonce i od těch s uzavřeným zdrojem. Neo využívá hierarchickou strukturu a umožňuje tedy převod do různých formátů jako HDF5 nebo MATLAB .mat. [18] [3]



Obrázek 2.4: Diagram zpracování

2.3 Proces zpracování dat

Způsob zpracovávání dat z EEG signálů není v současné době standardizován, a tak se můžeme setkat s různými principy a postupy zpracovávání. Při běžné práci s P300 signály jsou vstupní data různě předzpracovávána (např. převod reprezentace dat na vhodný formát pro práci s daty) nejen před začátkem experimentu, ale i v jeho průběhu.

Preprocessing je velmi důležitý v rámci celé sekvence zpracování. V rámci preprocessingu se zabýváme vhodnou reprezentací vstupních dat, extrakcí příznaků nebo různými ořezy či korekcemi. Tyto postupy jsou důležité proto, aby byla data správně interpretována.

2.3.1 Předzpracování dat

Před zpracováním a rozdělením dat (trénovací a testovací) je třeba data připravit. Použít lze filtrace, segmentace a baseline korekce.

Filtrace dat

EEG zachytí činnost mozku jako signál sinusového charakteru o různých frekvencích a fázích. Část filtrace dat si můžeme představit jako jednotky, které změní charakteristiku EEG signálu. K tomu využijeme frekvenční filtry, jimiž můžeme vybrat frekvence, které zůstanou nebo které budou potlačeny. Nejčasteji se používají filtry low-pass, high-pass a bandpass filtr. Low-Pass filtr tlumí vysoké frekvence, high-pass filtr tlumí nízké frekvence a bandpass filtr propouští jen frekvence určeného frekvenčního rozsahu. Při aplikaci filtrů je důležité dbát na správné používání. Hrozí riziko poškození dat. [4]

Neméně důležitými filtry jsou pásmově selektivní filtry. IIR (Infinite Impuls Response) je filtr, který má nekonečnou impulzní charakteristiku. Při implementaci je vyžadován rekurzivní přístup. Protože má tento filtr nelineární fázi, musí být počítáno s deformací křivky. FIR (Finite Impuls Response) je filtr s konečnou impulsní charakteristikou. Při implementaci proto nemusí být postupováno rekurzivně. Tento filtr je stabilní. Protože má lineární fázi, nedeformuje křivku. [4]

Segmentace dat

Při segmentaci je signál rozdělen na menší části, které mají podobné znaky. Jednotlivé segmenty vykazují specifické vlastnosti (minimum, maximum,

medián hodnot). Při členění signálu na jednotlivé segmenty je důležité zvolit vhodný postup, aby vznikl segment s požadovanými vlastnostmi. Jedním druhem jsou například segmenty o konstantní délce (například sekundy). Takové segmenty obsahují více amplitud. Proto jsou často v praxi často využívány. Další variantou je možné využití typu segmentace podle artefaktů. Může být tedy využito segmentace například podle amplitud. Rizikem segmentace je posun amplitud. Takto upravená data nejsou vhodná pro průměrování. [20]

Baseline korekce

Jak již bylo uvedeno, zvláště při neinvazivním záznamu činnosti nervové soustavy, je výsledný EEG signál ovlivněn řadou vnějších i vnitřních rušivých jevů, které se v signálu často projevují jako výchylky. Jejich korekce může být provedena pomocí *pre-stimulus* intervalu (až 200 ms před stimulem). Korekce se realizuje jako odečet naměřeného napětí od každé hodnoty segmentu. Po odečtu se normalizují vysoké výchylky napětí. [21]

2.3.2 Rozdělení dat a klasifikace

Vstupní data jsou po předzpracování následně rozdělena na trénovací a testovací.

Trénovací dataset

Trénovací data se využijí v rámci trénování klasifikátoru s učitelem. Je tedy předpokládáno, že je předem označeno, do jaké kategorie mají být zařazena. Klasifikátor si zkontroluje, jak byl v určení úspěšný, analyzuje chyby, a tím se zdokonaluje (učí).

Testovací dataset

Testovací datasety se využijí v rámci testování naučeného klasifikátoru. Hlavním účelem těchto dat je ověření správnosti naučeného klasifikátoru. I zde je předpokládáno, že jsou vstupní data předem označena.

Klasifikátor

Klasifikátor je za pomoci trénovacích dat naučen rozpoznávat zadané subjekty. Ověření je následně prováděno na základě testovacích dat.

2.3.3 Strojové učení

Jednou z nejvíce se rozvíjejících oblastí informatiky je umělá inteligence. Uplatňuje se při řešení komplexních úloh nebo při zpracování velkého objemu dat.

Strojové učení je jednou z podoblastí umělé inteligence. Zkoumá techniky a algoritmy, které umožní počítačovému systému „učit se“, tedy změnit se tak, aby zefektivnil schopnost přizpůsobení se změnám okolního prostředí.

Podle způsobu učení můžeme algoritmy strojového učení rozdělit do těchto kategorií:

Učení s učitelem (supervised learning)

Je jednou z nejvíce používaných forem strojového učení. Algoritmus se trénuje na označených datových sadách. Pro vstupní data je určen správný výstup. Taková množina dat s označením se často připravuje ručně nebo jen částečně automaticky, což je hlavní nevýhoda tohoto algoritmu. Během procesu učení se hodnotí správnost jednotlivých výstupů. Následně systém upraví své nastavení s cílem snížení chybovosti. Po trénování se výkon nastaveného systému ověří na jiné testovací množině. Ta slouží k otestování obecných schopností nastaveného systému a výkonu na datech, s kterými se systém ještě nesetkal.

Učení bez učitele (unsupervised learning)

Systém třídí a klasifikuje data na základě vzorů, které sám rozpozná, ke vstupním datům není známý výstup. Systém si vzory třídí do skupin a reaguje na typického zástupce, nebo si přizpůsobí topologii vlastnostem vstupu.

Kombinace učení s učitelem a bez učitele (semi-supervised learning)

Část vstupních dat má známý výstup, ale další data (většinou větší část) výstup nemá.

Zpětnovazební učení (reinforcement learning)

Algoritmus nepoužívá datové sady, ale informace, které získává z prostředí. V průběhu procesu učení se vyhodnocují a případně upravují zvolené strategie.

2.3.4 Neuronové sítě

Jedním z používaných modelů strojového učení jsou neuronové sítě. U běžných výpočetních systémů program začíná prvním řádkem kódu a po jeho vykonání následuje další. Neuronové sítě však neběží lineárně, ale paralelně ve všech uzlech. Pro mnoho typů strojového učení jsou zapotřebí strukturovaná data. Neuronové sítě jsou schopné interpretovat události v okolním světě jako data s možností zpracování. Dokáží zpracovávat velké objemy nelineárních dat a řešit složité problémy, které by jinak vyžadovaly zásah člověka. [6]

Umělá neuronová síť je inspirovaná fungováním biologických neuronových struktur. Skládá se z umělých neuronů, které jsou vzájemně propojeny a navzájem si předávají signály. Neuron je specifický tím, že má více vstupů, ale pouze jeden výstup. Výstup z jednoho neuronu může být vstupem pro další neuron.

Často se setkaváme s tzv. hlubokým učením (deep learning). Při něm se uplatňuje více vrstev, které jsou umístěny ve struktuře sítě. Hluboké učení sítí se týká neomezeného počtu vrstev omezené velikosti. Při hlubokém učení je povoleno, aby byly vrstvy heterogenní a aby se z důvodu efektivity, trénovatelnosti a srozumitelnosti sítě značně odchýlily od podoby biologických neuronových sítí. [6]

O výběru určitého typu neuronových sítí rozhodují požadavky na zpracování dat:

Konvoluční neuronové sítě (CNN)

Konvoluční neuronové sítě obsahují konvoluční vrstvy (konvoluce je matematický operátor zpracovávající dvě funkce). Jejich klíčovou vlastností je, že umí detekovat prvky snímku, jako jsou světlé či tmavé (nebo konkrétně zbarvené) body, okraje v různých orientacích, vzory apod. Tyto sítě se využívají při převodu obrazu na digitální matice, pro rozpoznávání známých obrázků a objektů. Využívají se pro detekci známých obličejů nebo analýzu zabarvení. [6]

Generativní adversariální sítě (GAN)

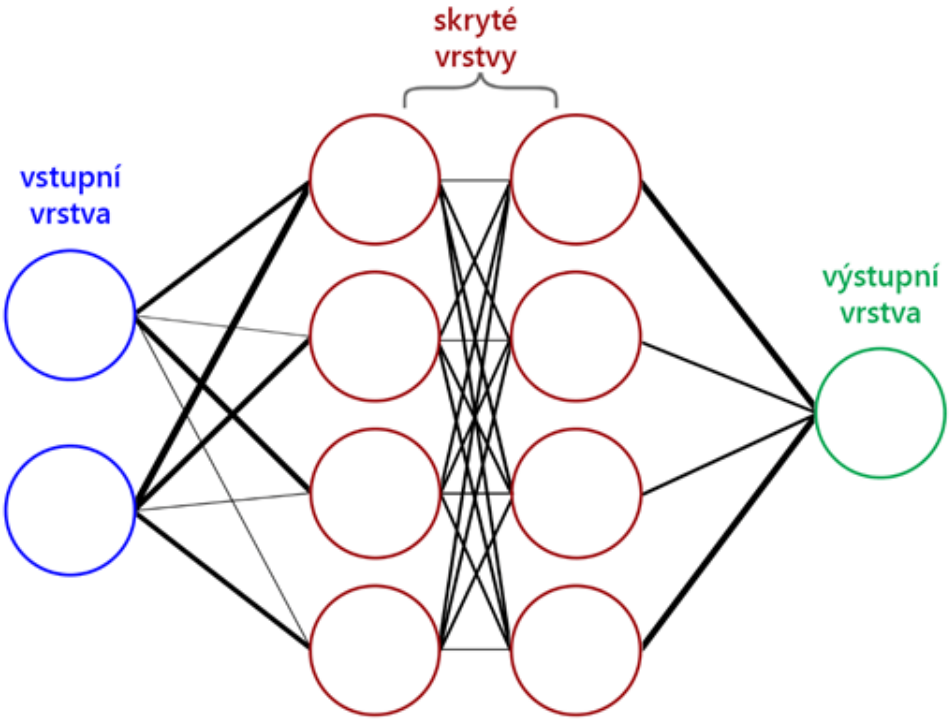
Tyto sítě využívají nejméně dvou systémů umělé inteligence, které se učí rychleji, než kdyby systém pracoval sám. Jeden systém pracuje na úkolu, druhý systém jeho výsledky hodnotí a poukazuje na případné chyby. Oba systémy spolu neustále komunikují, čímž se systém vylepšuje. [6]

Rekurentní neuronová síť (RNN)

Rekurentní neuronové sítě předávají data s určitou časovou prodlevou. Při výpočtech se uplatňují historické informace v každém specifickém stavu. Tyto sítě najdou uplatnění při rozpoznávání řeči, v robotice nebo při pokročilém prognózováním (finančních trhů nebo při vývoji pandemií). [6]

Transformátory

Tyto sítě se uplatňují při zpracování sekvenčních vstupních dat. Umožňují přidat různým vstupním datům různou míru vlivu. Tím je možné zkrátit dobu potřebnou pro natrénování modelu za použití metod paralelizace. [6]



Obrázek 2.5: Neuronová síť, zdroj: [13]

2.4 Augmentační techniky

Signál z elektroencefalografu může být zpracován několika způsoby. Při výběru je důležité dbát na povahu dat. Je potřeba volit vhodné techniky, případně jejich kombinace. Metody můžeme nadále rozdělit na ty, u kterých se můžeme vrátit do původního stavu, když známe velikost změny (posun), a metody, u kterých je obtížnější rekonstrukce původních dat (šum). [24] [16]

2.4.1 Přidávání šumu

Šum je jednou z běžných nedostatků ve vstupních datech. Pomocí vhodně zvolené konvoluční matice můžeme vstup aproximovat pomocí signálů podobných vstupu. Velikost je určena na základě odstupu signálu a šumu. Při nevhodně zvoleném poměru fakticky signál zmizí a zůstane jen šumová stopa. Tento stav však není žádoucí. Vhodně zvolený poměr není možné určit pomocí dostupných emperických vzorců. Hodnotu je třeba vybrat a vyzkoušet na různých vstupních datech. Je tedy zřejmé, že se liší experiment od experimentu. Někdy může docházet k zvýraznění extrémních hodnot. Pomocí extrémů můžeme zkoumat jisté statistické jevy. Šum také pomůže diverzifikovat vstupní signál. Díky extrémům je možné rozlišovat v rámci signálu různé stavy. [25]

Během snímání dat se setkáváme s různými prostředky rušení. Šum řadíme mezi jeden z nich. Přidáváním šumu do dat můžeme zvýšit schopnost klasifikátoru reagovat i na data, která obsahují tuto nežádoucí složku. Cílem je, aby klasifikátor dokázal správně zpracovat i data ovlivněná šumem.

2.4.2 Změna rychlosti

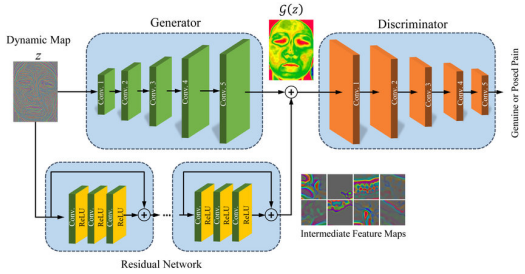
Jedná se o změnu, která ovlivní rychlost průběhu výsledného signálu. Rizikem při zrychlení může být ztráta přesnosti a odlišitelnosti jednotlivých jevů, které se odehrávají v daném úseku. Vstupní data mohou být také zpomalena. I zde hrozí jisté problémy. Při příliš vysokém zpomalení můžeme narazit na přesnost záznamu, tedy bude nutné úseky interpolovat.[25]

2.4.3 Posun

Tímto způsobem je možné posouvat vstupní data o libovolný úsek. Jistým způsobem můžeme brát posun i jako ořez - vybereme jen tu nejpodstatnější část z celku. Posuneme-li střed vstupních dat do počátku, máme následně jen jistou vybranou část. Tato metoda je snadno dosažitelná pomocí interních knihovních funkcí *MNE*. [25]

2.4.4 GAN sítě

GAN sítě, nebo-li Generative adversial network, jsou speciálním typem strojového učení, který se snaží vygenerovat statisticky podobná data, jaká získal na vstupu (tedy z trénovací množiny dat). Idea generativních adversariálních neuronových sítí je postavena na nepřímém učení prostřednictvím diskriminátoru. Princip jejich fungování si můžeme představit jako dvě proti sobě



Obrázek 2.6: GAN síť [31]

soupeřící neuronové sítě. První síť se specializuje na generování objektů dle naučeného modelu, druhá kontroluje naopak jejich věrnost a reálnost. Narozdíl od ostatních metod se zde nesnažíme hledat největší podobnost (resp. minimalizovat vzdálenost ke specifickým objektům), ale učíme se generovat data tak, aby prošla přes kontrolní neuronovou síť. [25]

2.4.5 SMOTE

SMOTE (*Synthetic Minority Oversampling Technique*, zkráceně SMOTE) je speciální technika určená pro augmentaci nevyvážených datových souborů. Často se setkáváme s daty, v nichž zastoupené třídy jsou nevyváženy. Nevyváženost způsobí horší výsledky učení (accuracy). Řešením je tzv. „dovzorkování“ méně zastoupené třídy na úroveň třídy majoritně zastoupené. [29]

3 Metriky měření úspěšnosti

Při provádění experimentů strojového učení je důležité mít možnosti, jak získané výsledky ohodnotit. K tomuto účelu se používají tzv. *metriky*. Tyto ukazatele popisují míru úspěchu klasifikace. Vhodnost užití jednotlivých metrik se však diametrální liší v závislosti na stanovém experimentu a vstupním datovém souboru.

3.1 Samostatné metriky

Samostatné metriky jsou taková ohodnocení, která nejsou složena z více jednotlivých metrik. Výpočet je tedy jednodušší. Naopak využití v některých případech není doporučováno.

3.1.1 Accuracy - přesnost

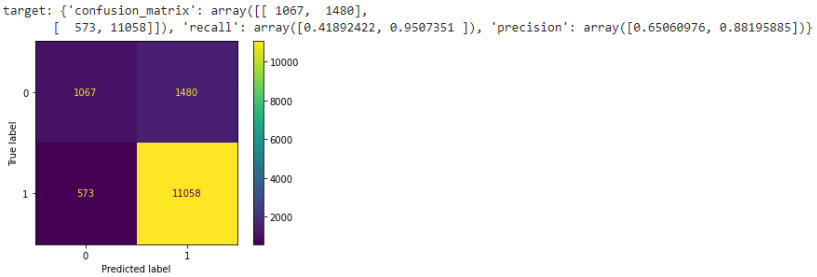
Accuracy je jednou z nejjednodušších metrik, kterou si můžeme představit. Udává fakticky procentuální úspěšnost rozpoznávání na testovacích datech. Mějme například testovací množinu o velikosti 100 objektů. Klasifikátor nám rozpozná 93 objektů. Přesnost je tedy 0,93 nebo-li 93%.

3.1.2 Precision

Accuracy je vhodná, pokud jsou jednotlivé třídy objektů zastoupeny rovnoměrně v rozpoznávané množině. Proto kdyby fakticky všechny testovací objekty rozpoznal jako nejčetnější zastoupenou třídu, byla by accuracy stále relativně vysoká, ale nebylo by možné z výsledků cokoliv vyčíst. Precision se zde definuje jako poměr úspěšně rozpoznaných objektů dané třídy vůči všem objektům, které klasifikátor rozpoznal jako tuto třídu.

3.2 Kombinované metriky

Kombinované metriky, jak již z názvu vypovídá, jsou kombinací dříve zmíněných metrik. Výhoda těchto metrik se projeví například při nevyváženém datovém souboru.



Obrázek 3.1: Confusion matrix, zdroj: vlastní projekt

3.2.1 Confusion matrix - matice záměn

Matice záměn je jednou z často používaných metod v kontextu klasifikačních úloh strojového učení. Sloupce matice odpovídají skutečné hodnotě předpovídaného znaku. V řádku jsou předpovědi klasifikátoru. Na diagonále tedy najdeme hodnoty správně určených prvků, mimo ni jde o chyby. Pomocí výsledků se nechají lépe určit vlastnosti a kvalita klasifikátoru.

Často můžeme matici záměn vidět v grafické podobě, kde jsou hodnoty znázorněny pomocí různých barev. Příklad grafického znázornění najdeme na obrázku 3.1.

3.2.2 Precision - Recall

Precision (přesnost) a recall (výtěžnost) pokrytí zkoumané metody se začíná využívat u klasifikačních úloh a machine learningu v poslední době. Přesnost si představme jako poměr relevantních výsledků analýzy vůči všem výsledkům analýzou získaných. Výtěžnost si představme jako poměr relevantních výsledků analýzy vůči všem relevantním jevům. Precision a recall tedy přesněji popisuje věrnost dat oproti čisté precision nebo accuracy.

3.2.3 F - skóre

F skóre je jednou z metod statistické analýzy. Pro svůj výpočet využívá hodnot získaných pomocí **Recall** a **Precision**. Nejnižší možnou hodnotou, kterou může F-skóre nabývat, je 0,0. A naopak nejvyšší hodnotou je 1,0. Hodnota se vypočítává jako kombinace hodnot z předešlých metod.
Vzorec:

$$Accuracy = 2 * \frac{precision * recall}{precision + recall}$$

Výše zmíněné metriky se při experimentech zaznamenávají do classification reportů, nebo-li zpráv o klasifikaci.

4 Popis experimentu a analýza dat

Tato bakalářská práce vychází z projektu studenta Fakulty aplikovaných věd ZČU v Plzni Romana Kalivody, jehož klasifikátor je použit jako referenční pro zjištění výsledků měření (případně zlepšení). Kompletní množina souborů potřebných pro běh experimentu byla poskytnuta vedoucím bakalářské práce Ing. Pavlem Šnejdarem.

Cílem našeho zkoumání je najít počáteční příznaky demence. Pro tento účel využíváme audiovizuální data, která svým průběhem připomínají reakci osoby s počáteční demencí. Naše zkoumané podněty se vyskytují v rámci P300 dat tedy výhradně v rámci TARGET stimulů. Jelikož je dodaný datový soubor rozsahem připraven na větší experimenty, nejsou využita všechna data. V rámci našeho experimentu byla využita jen audio, audiovizuální a vizuální část. Tlačítko (button) by bylo využito pro potřeby měření reakce mezi stimulem a reakcí ruky.

4.1 Vstupní data

Vstupní data představují záznamy elektroencefalografu zdravých jedinců. Zaznamenána je jejich reakce na smyslový stimul (vizuální, audio, audiovizuální).

V této kapitole jsou charakterizována vstupní data a uvedeny možnosti jejich zkreslení. Elektroencefalograf zaznamenal reakci na různé typy stimulů. Zaznamenána byla data z měření celkem 16 osob. Můžeme pracovat s daty naměřenými různými sondami i zvlášť oddělenými záznamy reakcí na různé typy stimulů.

4.1.1 Charakteristika vstupních dat

Data získaná při běžných EEG vyšetřeních mohou být zkreslena řadou netechnických faktorů. Výsledky může ovlivnit únava osoby (například při nedostatku spánku), užívané léky, nápoje obsahující povzbuzující látky (například kofein nebo tein) nebo jídla obsahující glutamát sodný či vitamín B. Při experimentálním měření byly pro všechny subjekty zajištěny stejné podmínky, data by tedy měla být minimálně zkreslena. Další technické ovlivňující kvalitu vstupních dat jsou uvedeny v kapitole 2.2.1.

Zkoumán byl EEG signál z několika sond (Pz, Cz). Měřena byla rychlost reakce nervové soustavy na daný stimul. Uvedená data nejsou komplexní, vybrána byla jen ta, která jsou dostatečná pro ověření hypotézy a experimentu (úprava a zpracování dat IT technologiemi).

4.1.2 Obsažené stimuly

Při EEG vyšetření byla zaznamenána reakce mozku na různé stimuly. Měřena byla především rychlost reakce nervové soustavy.

Nejrychlejší odezvu pozorujeme u vizuálních podnětů. Data z těchto měření představují reakci zdravého jedince. Data audiovizuální s delší dobou odezvy v experimentu představují reakci člověka s počínající demencí. Nejpomalejší reakce nervové soustavy je zaznamenána při audio stimulu. Data z těchto měření představují pacienta s rozvinutou demencí.

Tímto postupem můžeme simulovat různé fáze demence. Na základě toho lze zkoumat a hodnotit jednotlivé metody zpracování EEG signálu, zlepšit klasifikace a tím lépe diagnostikovat počáteční demenci.

4.1.3 Velikost a formát dat

Vstupní data jsou uložena v *.MAT* souborech. MAT soubory umožňují rozdělit ukládaná data do tzv. "setů". Ani vstupní data nejsou výjimkou. Každý ze setů může mít uloženy také metainformace o uložených datech.

Uvnitř najdeme naměřená EEG data z jednotlivých sond. Celkem se jedná o data od 16 subjektů. Soubory jsou rozděleny do složek dle druhu stimulu (visual, audiovisual, audio). V rámci každé z těchto složek jsou pro každý měřený subjekt 4 soubory (*train*,*test_1*,*test_2* a *button*).

Datový soubor obsahuje signál z těchto sond: Cz, CPz, POz, Pz P1, P2, C3, C4, O1, O2, T7, T8, P3, P4, F3, F4. Tyto elektrody jsou zaznamenány jako jednotlivé dimenze `numpy` pole.

4.1.4 Vyvážený a nevyvážený datový soubor

Při práci s datovými soubory se můžeme setkat s již vyváženou variantou nebo nevyváženou (imbalanced dataset). Nevyvážený datový soubor znamená, že data zařazená do jednotlivých kategorií (v tréninkové fázi strojového učení) nejsou rovnoměrně rozdělena, v některých kategoriích (třídách) je více dat než v jiných.

Nevýhodou nevyváženého datového souboru jsou rizika při využíváni metrik úspěšnosti. Využití accuracy jako metriky měření úspěšnosti klasifikace

může vést k dezinterpretaci výsledků. Výrazně vyšší zastoupení jedné z kategorií by mohlo vést při absolutním zařazování do této kategorie k relativně pozitivním výsledkům. Řešením tohoto problému mohou být parciální klasifikace po jednotlivých třídách. Významným způsobem zvýšené zastoupení jedné třídy může upozornit na vzniklé problémy včas. Proto je vhodné pro nevyvážený datový soubor volit spíše jiné metriky nebo pomocí speciálních technik rozšířit méně zastoupené třídy.

4.2 Analýza vstupních dat

Na vstupu jsou očekávány soubor *.MAT* z `bci-classificatoru`. Poměr mezi testovacími a trénovacími daty z hlediska velikosti souborů je 24250:11500 eventů. Trénovací data jsou zároveň vstupem pro generátor rozšířené datové sady. Pomocí trénovacích a testovacích dat validujeme následný generátor vůči bci-classificatoru.

4.2.1 Struktura vstupních .MAT souborů

Pro natrénování klasifikátoru je využíváno *.MAT* souborů. Jedná se o soubory programu MATLAB. Data v souborech můžeme rozdělit do několika skupin: nosná data a metadata.

Nosná data obsahují samotné naměřené hodnoty z sond elektroencefalografu a další přidružené informace (například labely, subsety). Metadata obsahují informace o třídách, setech (train, test) nebo o rozmístění sond a časovém výskytu jednotlivých stimulů.

MAT soubory mohou být využívány i pro jiná data než z EEG. Vnitřní struktura těchto souborů může být tedy rozdílná.

V našem případě jsou data rozdělena do 7 podpoložek. Každá položka je důležitá pro zpracovávání. Informace musí obsahovat jak metadata (časové rozdělení, elektrody, atd.), ale také samotná naměřená klíčová data.

Položky .MAT souborů:

- `allTARGETS` Naměřená target data
- `allNTARGETS` Naměřená non-target data
- `__header__` Hlavička .MAT souboru popisující platformu a verzi MATLABu

- `__version__` Version udává verzi dat
- `__globals__` Global odkazuje na globální nastavení
- `electrodes` Seznam použitých elektrod
- `tSCALE` Časová osa zaznamenaných podnětů

`allTARGETS` obsahuje veškerá naměřená *TARGET* data. V rámci generování je tato množina použita jako vstupní pro generování příslušných *TARGET* dat.
`allNTARGETS` obsahuje *NON-TARGET* data, která opět byla použita jako vstup pro naučení neuronové sítě.
`header` obsahuje hlavičku popisující verzi `MATLABu`, dále platformu a datum vytvoření
`version` obsahuje verzi snímaných dat, případně může obsahovat další metainformace potřebné pro sběr
`electrodes` popisuje název a rozmístění snímaných elektrod
`tSCALE` označuje rozložení jevů na časové ose, z těchto dat se počítá perioda a vzorkovací frekvence

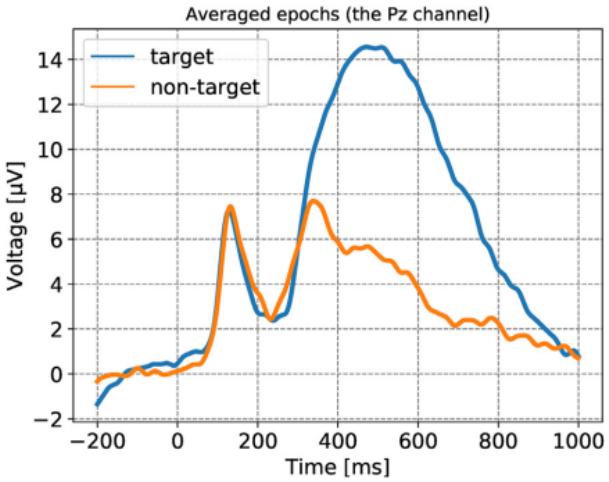
Data jsou v rámci Pythonu interně reprezentována a ukládána jako dictionary. Ukládání zajišťuje metoda *savemat* z knihovny *scipy.io*. Realizace v rámci pythonu je v Ukázce kódu 1.

```
electrodes = np.array([['Cz'], ['CPz'],
                           ['POz'],['Pz'],
                           ['P1'], ['P2'],
                           ['C3'], ['C4'],
                           ['O1'], ['O2'],
                           ['T7'], ['T8'],
                           ['P3'], ['P4'],
                           ['F3'], ['F4']])
data = {"allTARGETS": data_target,
        "allNTARGETS": data_ntg,
        "__header__": "'MATLAB 5.0 MAT-file, Platform:
                MACI64, Created on:Wed Jul 8 11:55:27 2015'",
        "__version__": "1.0",
        "__globals__": np.array([]),
        "electrodes": electrodes,
        "tSCALE":  np.array([[-1.9807e-01,
                  -1.9602e-01 ... ]])}
```

Ukázka kódu 1: Popis struktury .MAT souborů

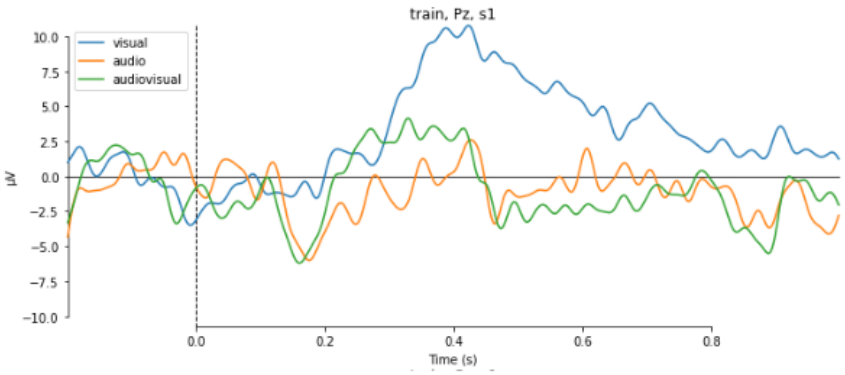
4.2.2 Target a non-target stimuly

Cílové a necílové stimuly (target, non-target) jsou v datech jasně patrné. V rámci P300 vln se amplituda zvýší, vyskytuje-li se cílový stimul po větším počtu necílových stimulů. Většinou bereme non-target vstupy jako měření, které proběhlo v době, kdy nebyl subjekt soustředěn. Taget naopak odpovídá situaci, kdy bylo měření vykonáváno za plného soustředění zkoumaného jedince. Nebo target stimul je takový stimul, na který bude testovaný subjekt upozorněn ještě před začátkem experimentu. [14]



Obrázek 4.1: Srovnání TARGET a NON-TARGET epoch (zprůměrovaných). Dle předpokladů je vlna P300 zastoupena v rámci TARGET stimulu. [32]

Na následujícím obrázku jsou vizualizovány zprůměrované vlny. P300 je díky tomu jasně viditelná.



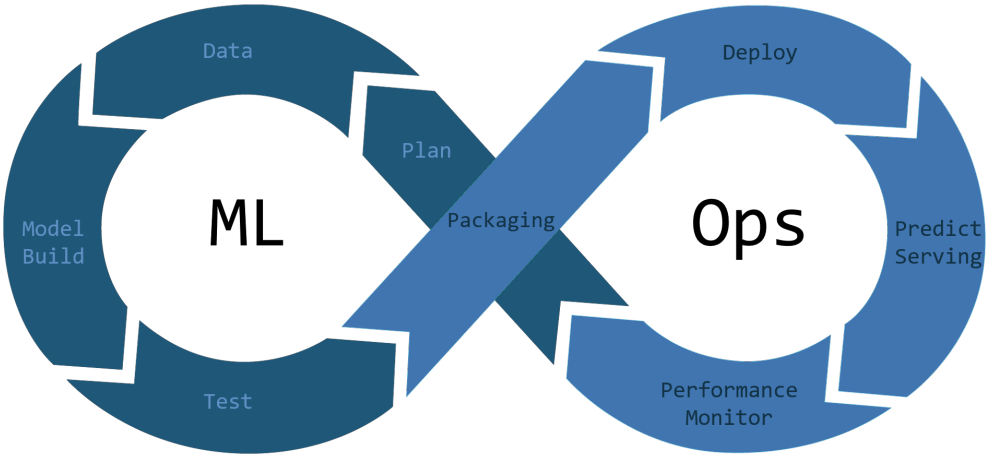
Obrázek 4.2: Vizualizace P300 v rámci A,V,AV stimulů

5 Návrh a implementace řešení

Výsledky strojového učení jsou přesnější, pokud máme k dispozici větší množství vstupních dat. Stroj se, stejně jako lidé, zlepšuje díky větší praxi. Cílem této práce bylo právě bylo navržení a rozšíření vstupního datového souboru. Pro lepší představu a pro pochopení popsané problematiky uvádím i příklady kódových segmentů.

5.1 Technologie využívané při strojovém učení

Technologie, které v současné době plně pokrývají jistou oblast IT, se nazývají obecně XOps (například DevOps - vývojáři). V případě strojového učení tomu není jinak. MLOps (machine learning) je trend současné doby, který popisuje celý životní cyklus projektu zaměřeného speciálně na oblast strojového učení. Životní cyklus vývoje naznačuje následující diagram 5.1. Jde o neustále se opakující cyklus.



Obrázek 5.1: MLOps, zdroj: [8]

Při zakládání projektu je důležité plánování, které přispěje k lepší organizaci samotné práce. Následuje analýza dat. I v případě této bakalářské práce byla důležitá (struktura *.MAT* souborů pro správné ukládání dat na

disk). Po analýze přichází na řadu tvorba samotného modelu. Někdy (například v této práci) se tvoří dokonce 2 modely (generátor, diskriminátor). Důležité však je, aby vytvořené modely správně fungovaly. Správnou funkci modelu proto musíme vždy otestovat. Pokud již víme, že program funguje dle představ a zadání, je načase pro model vytvořit balíček a distribuovat jej. I následně je důležité sbírat jednotlivé metriky a podle nich plánovat další rozvoj. A zde se tedy celý životní cyklus projektu opět vrací na začátek.

5.1.1 Jupyter Notebook

Cílem vývojářů Jupyter Notebooku z Project Jupyter bylo vytvořit open-source software, který umožní interaktivní výpočty pro mnoho programovacích jazyků. Název tohoto projektu je odvozen od jazyků, které Jupyter Notebook umí zpracovávat, a to: python, R, Julia. V rámci projektu byly doprogramovány pomocné nástroje pro jednodušší práci s vyvinutými technologiemi.

Jupter Notebook je webová technologie pro vytváření "notebooků". Každý jeden projekt je složen z buněk, které se mohou spustit samostatně, mohou být spuštěny všechny naráz nebo pomocí speciálně doprogramovaného RUNNERU může být spouštění řízeno zvenčí. Runner navíc umožní spouštět projekt s pomocí předem definovaných parametrů a tím může ovlivňovat běh prováděných experimentů. Ve spojení s MLFlow je možné uchovávat výsledky experimentů v externím úložišti a sledovat případný vývoj.

5.1.2 Google Colab

Google Colab je volně dostupný nástroj pro cloud computing od společnosti Google. Nástroj je dostupný všem pomocí webového prohlížeče. Uživatelské rozhraní tohoto systému je odvozeno od technologie Jupyter Notebooku. Výhodou této technologie je plné propojení s technologiemi od společnosti Google (Google Drive), které lze využít například pro ukládání jednotlivých Jupyter Notebooků, trénovacích a testovacích dat, vygenerovaných souborů nebo natrénovaných modelů. Navíc umožňuje pro práci využít virtualizovanou grafickou kartu pro urychlení výpočtů. Google Colab umožňuje práci jak s Pythonem 2, tak i s Pythonem 3. Google umožňuje využívat různá rozšíření, například Google tensor processing unit pro urychlení jednotlivých výpočtů (například v kooperaci s Julia on Notebook).

5.1.3 Python

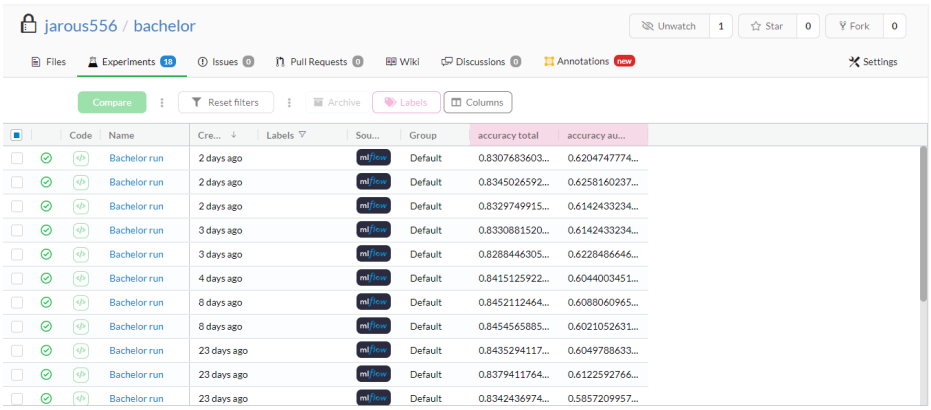
Python je vysokoúrovňový programovací jazyk navržený Guidem van Rossumem roku 1991. Python vyniká svou jednoduchostí. Nabízí programátorovi dynamickou kontrolu datových typů a podporuje různá paradigmata (objektově orientované programování, procedurální nebo funkcionální programování). Poslední dobou patří mezi nejoblíbenější jazyky (za březen 2022 je nejoblíbenějším programovacím jazykem vůbec – dle TIOBE indexu).

5.1.4 MNE Tools

MNE Tools je velmi užitečný open source balíček v Pythonu, který se stará o vizualizaci, zpracovávání a analýzu neurofyziologických dat jako EEG, MEG, sEG nebo ECOG. Pomocí MNE Tools můžeme transformovat EEG signál (dopředu, dozadu), dále můžeme využít znalosti sond, vizualizovat sondy na hlavě, případně zobrazit spolu se sondami i příchozí signál. Mezi výhody řadíme možnost využití více vstupních formátů dat.

5.1.5 MLFlow

MLFlow je open source technologie, která pomáhá machine learning specialistům s managováním životního cyklu machine learningu, se zpracováváním jednotlivých experimentů z centrálního místa, s reprodukcí jednotlivých modelů a výsledků, s distribucí naučených modelů a také ukládáním jednotlivých naučených modelů a jejich vstupních a výstupních datových souborů spolu s kódem aplikace.



Obrázek 5.2: MLFlow instance, zdroj: vlastní projekt

5.1.6 Prostředí DAGsHub.com

Prostředí **DAGsHub.com** poskytuje veškeré potřebné nástroje typu MLOps, které jsou potřebné při strojovém učení. DagsHub je online nástroj poskytovaný společností DAGsHub. V rámci jedné instance nalezneme nástroje typu Git pro verzování kódu, lokální wikipedii pro vedení dokumentace v jazyce Markdown, nástroje pro správu modelů, metrik a datasetů (nástroj DVC) nebo je možné pomocí různých vizualizací reprezentovat schéma zpracování dat (nebo-li pipeline) strojového učení.

5.1.7 Docker

Docker je open-source nástroj pro izolaci aplikací do samostatných kontejnerů. Docker lze doinstalovat do prostředí MacOS, Linux i do Windows (jedná se o odlehčenou virtualizaci). Windows potřebuje pro chod kontejneru nainstalovaný Windows Subsystem for Linux. Běh kontejnerů se využívá v rámci strojového učení pro potřeby běhu nezávislých experimentů s vysokými nároky na hardware na společných výpočetních clusterech nebo superpočítačích.

5.1.8 DVC

Data Version control je nástroj pro verzování datových souborů, workflow, modelů a slouží jako experiment management software. Pro fungování využívá v základu Git. Díky propojení s různými technologiemi (např. portál DagsHub.com) může být tato technologie využita i pro jiné účely (verzování souborů metrik). Technologii je možné používat v rámci všech dostupných operačních systémů (MacOS, Linux, Windows).

5.1.9 Keras

Keras je open-source knihovna pro vývoj neuronových sítí. Keras slouží jako rozhraní v rámci knihovny **TensorFlow**. Tato technologie obsahuje řadu modulů pro implementaci nejrozšířenějších neuronových sítí, dále obsahuje komponenty pro tvorbu jednotlivých skrytých vrstev, aktivačních funkcí a metrik pro zhodnocení míry učení klasifikátorů a neuronových sítí. Ve spolupráci s TensorFlow umožňuje vývojářům jednodušší načítání známých datasetů (*tf.data.Dataset*) využívaných v rámci PoF (proof-of-concept).

5.2 Návrh řešení

Návrh řešení vychází ze znalosti zvolených technologií.

5.2.1 Výběr augmentačních technik

Mezi augmentační techniky pro rozšíření datového souboru patří například: posun, šum, změna rychlosti, GAN sítě nebo SMOTE.

Na základě časové náročnosti a charakteristik byly zvoleny následující techniky:

- Posun
- Šum
- Kombinace předchozích
- GAN sítě

Z důvodu nízké časové náročnosti na implementaci bylo zvoleno použití technik založených na posunu. Šum byl zvolen proto, že často provází většinu měřených dat. Volba augmentace šumem pomůže zvýšit robustnost klasifikátoru. Augmentace prostřednictvím GAN sítí byla zvolena, protože dosahuje nejlepších výsledků [23]. Na základě konzultace s vedoucím práce byla přidána ještě kombinace metod posun a šum. Technologie SMOTE a změna rychlosti nebyly vybrány z časových důvodů.

5.2.2 Výběr metrik měření úspěšnosti

Po nastudování jednotlivých metrik jsem zvolil metriky, které byly pro bakalářskou práci nejvhodnější. Výběr probíhal z metrik představených v rámci kapitoly 3 - těmi jsou: accuracy, precision, confusion matrix, precision-recall a F-skóre.

Pro měření úspěšnosti byly zvoleny následující metriky:

- Precision
- Accuracy

Na základě [27] jsem se rozhodl využít nejčastěji používané metriky měření úspěšnosti. Těmito metrikami jsou výše zmíněné: accuracy, precision. Volba byla dále ovlivněno možností jednoduchého sledování vývoje zmíněných metrik pomocí příslušných nástrojů (MLFlow v našem případě).

5.2.3 Výběr typu experimentů

Výběr typu experimentů (množství generovaných dat) byl ovlivněn typem augmentace datového souboru. Maximální množství různých vygenerovaných souborů je u některých metod augmentace ovlivněno principy fungování metody. Proto bylo u těchto metod využito maximálního potenciálu (posun - přidání 100%). U metod šum a kombinace byly na doporučení vedoucího realizovány shodné poměry přidávání. Velikost přidávání u GAN sítí byla stanovena na základě článku [23].

Pro GAN sítě byly realizovány tyto experimenty:

- Přidání 50%
- Přidání 100%
- Přidání 150%

Bližší zdůvodnění je v rámci sekce 5.5.4

5.2.4 Výběr technologií projektu

Po prostudování jednotlivých technologií byly zvoleny prostředky vyhovující potřebám bakalářské práce.

Na základě zadání bakalářské práce byl zvolen programovací jazyk Python. Pro běh jednotlivých notebooků byl nejdříve využíván Google Colab, následně se projekty přesunuly do prostředí Jupyter Notebook, které disponovalo větším množstvím paměti RAM díky provozu na lokální stanici. Pro práci práci s EEG daty bylo využito knihovny MNETools. Volba frameworku pro potřeby učení neuronových sítí byla provedena na základě dřívějších zkušeností s knihovnou Keras. Pro záznam dat bylo využito technologie MLFlow. Volba MNETools a MLFlow proběhla na základě doporučení vedoucího práce. Pro běh MLFlow bylo po průzkumu možných cloudových technologií vybráno řešení portálu DAGsHub.com, který nabízel výrazně větší množství doplňkových služeb oproti konkurenčním nástrojům (Databricks).

5.3 Proces rozšíření a jeho vlastnosti

Experiment jako takový je ovlivněn mnoha faktory. Důležitá je volba postupu měření a následného zpracování výsledků. Vhodné je uvést pro opakování experimentu možné vedlejší faktory, které mohou experiment ovlivnit.

Dalším vhodným bodem je soupis informací, které obsahuje naměřený datový soubor. Popis může být v externím dokumentu, vědeckém článku nebo přímo v datech (vhodným pojmenováním).

Vliv na experiment může mít stavba vstupních datových souborů. V některých případech si data zajišťujeme sami, v jiných jsou nám dodána. Při analýze dat a návrhu experimentu je vhodné brát v potaz strukturu dat. Je-li vstup vyvážený či nevyvážený (třídní zastoupení jednotlivých subjektů). Podle těchto metrik je následně vhodné zvolit strukturu experimentu.

Struktura a aplikovaný postup experimentu jsou nejdůležitější částí ze všech. Vhodná stavba experimentu může šetřit čas a vhodným způsobem vést k dobrým výsledkům. V případě experimentu je vhodné uvádět nejen strukturu zpracovávání dat, ale i popis průchodu datového souboru v rámci experimentu. Při mnohých experimentech vznikají rozšíření datové sady (například u nás). Je pak vhodné uvést sturkturu nově vzniklých dat a prerekvizity pro jejich vznik.

5.3.1 Postup měření a získávání výsledků

V rámci experimentů probíhala různá měření s různými vstupními daty. Na vstupu se tedy objevily vstupní datové soubory s různým procentuálním zastoupením dogenerovaných dat v různých vstupních kanálech (A,V,AV). Při měření byly pozorovány změny nejen u celkové accuracy (nebo-li přesnosti), ale také změny metrik u jednotlivých stimulů. Výsledky pochází z průměrování 5 naměřených hodnot. Měření probíhala nezávisle a byla spouštěna pomocí speciálního *runneru* (běhového skriptu), který zajišťoval automatické zaznamenávání hodnot do externího systému na měření metrik, tedy **MLFlow**.

Délka běhu GAN sítě pro generování dat byla závislá na tom, jak velké množství dat bylo generováno. Při běžném generování dat, při kterém bylo dogenerováno 100%, byla délka běhu sítě zhruba 60 minut. Při zvětšování procentuální velikost vygenerovávaných dat se postupně délka běhu GAN sítě zvyšovala s každými dalšími 100% o dalších 6-7 minut. Generování bylo velmi náročné na výpočetní prostředky, proto na generujícím zařízení nebylo možné během běhu experimentu vykonávat žádné další činnosti.

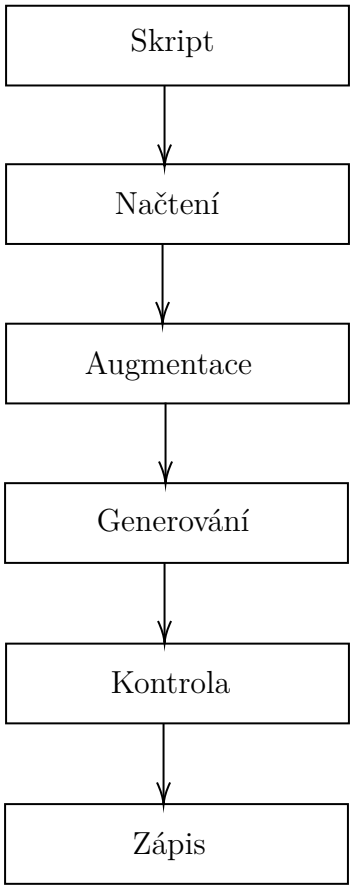
Měření bylo prováděno na pracovní stanici DELL (procesor: Intel Core i5 4300M 2.60 GHz, RAM: 16GB DDR3L, grafická karta: Intel HD Graphics 4600, SSD Kingston 250 MB - rychlost čtení/zápis: 500/450 MB/s) s grafickou kartou o nízké velikosti pracovní paměti. Experiment probíhal za použití výše zmíněné technologie `Jupyter Notebook`, `MLFlow`, `DAGsHub` a `Microsoft Excel` pro výsledné zpracování naměřených výsledků.

5.3.2 Schéma postupu zpracovávání

Schéma postupu zpracování (nebo-li pipeline) pro tuto práci obsahuje pipeline pro generování, která popisuje, jakým způsobem je generován nový datový soubor. Datová pipeline popisuje, jakým způsobem jsou zpracovávána vstupní data.

Pipeline pro generování

Generátor převzal vstupní množinu dat získanou z vědeckého projektu popsaného výše. Nad danou množinou provedl již zmiňované operace. Výstupem jsou nová vstupní data, která byla použita na vstupu klasifikátoru.



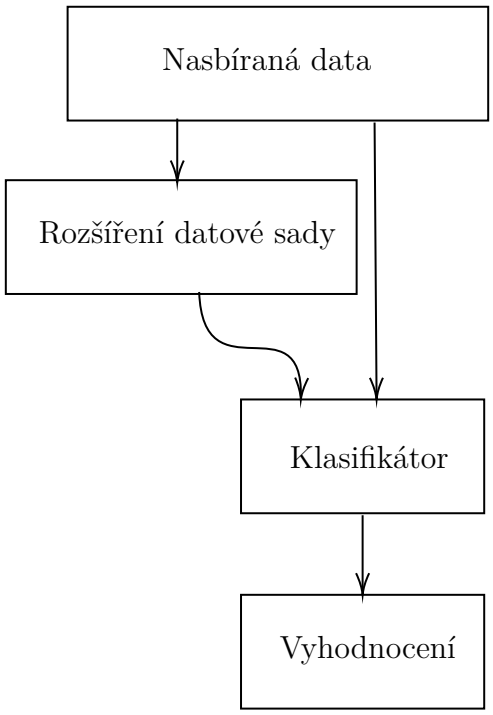
Obrázek 5.3: Pipeline

- Skript - skript se postará o přípravu běhového prostředí (import knihoven, přístup k souborům)
- Načtení - v rámci této části jsou vstupní data načtena do struktury

- Augmentace - úprava načtených dat pro generování
- Generování - data budou zpracována (posun, šum, GAN síť) a výsledné výstupní vzorky budou předány k vyhodnocení
- Kontrola - před zápisem na disk se vygenerovaná data zkontrolují
- Zápis - veškeré výstupy jsou zapsány na disk

Datová pipeline

Na vstupu jsou původní data, nad nimi jsou provedeny základní operace generátorem napsaným v jazyce Python. Výsledkem je rozšířený datový soubor.



Obrázek 5.4: Datová pipeline

- Nasbíraná data - data, která jsou již předzpracovaná, ořezaná a připravená pro rozšíření či pro vstup do klasifikátoru
- Rozšíření datové sady - skript, který pomůže rozšířit datovou sadu, aby byla robustnější

- Klasifikátor - data budou zpracována a výsledné výstupní vzorky budou předány k vyhodnocení
- Vyhodnocení - zhodnocení práce provedené klasifikátorem

5.3.3 Ladění hyperparametrů

Cílem ladění hyperparametrů je významným způsobem zvýšit metriky u aktuálního experimentu. Ladění se může zaměřovat na celý průběh experimentu (počet epoch, architektura modelu).
V rámci zkoumání se zabýváme těmito vlastnostmi sítí:

- počet skrytých vrstev
- počet neuronů v každé vrstvě
- míra učení
- aktivační funkce

Při ladění je důležité si stanovit parametry, které budeme měnit, a rozsah změn. V rámci ladění probíhá standardní učení, u kterého jsou následně na jeho konci výsledky zvalidovány a porovnány dle zvolené metriky.

Dále je důležité zvolit postup ladění jednotlivých metrik. Lepší je na začátku volit takové vlastnosti, u nichž víme, že budou mít vysoký vliv na zlepšení.

V případně naší práce jsem se zaměřil na zvyšování accuracy pomocí výše zmíněných metod. Pro realizaci byly vytvořeny 3 modely. Zkoumání bylo zaměřeno primárně na počet skrytých vrstev.

V rámci zkoumání byly provedeny experimenty s následujícími parametry:

- počet skrytých vrstev - 2
- počet skrytých vrstev - 3
- počet skrytých vrstev - 4

Testované scénáře byly prováděny na experimentu se 100% přidaných dat. Byly spouštěny pro každý typ neuronové sítě 3 experimenty a byla měřena celková accuracy (přesnost dat). Pro 2 skryté vrstvy a 4 skryté vrstvy nebylo dosaženo u celkové accuracy (přesnost) při zadané průměrné odchylce takových výsledků jako u 3 skrytých vrstev. V rámci experimentu proto bylo následně pokračováno se 3 skrytými vrstavmi.

Dosažené výsledky při 100% přidávání dat jsou zaneseny v rámci tabulky 5.1

Počet vrstev	Průměrná test_accuracy
2 vrstvy	0,84210
3 vrstvy	0,86721
4 vrstvy	0,85306

Tabulka 5.1: Porovnání počtu laděných vrstev

5.4 Architektura experimentu

Při práci s projekty je nutné dobře znát charakteristiky projektu. Mezi jednu z nejdůležitějších charakteristik patří zvolený programovací jazyk a příslušné knihovny. Dále je nutné znát i adresářovou strukturu projektu pro dobrou orientaci při práci.

5.4.1 Adresářová struktura

ářová struktura vychází z předpřipravené struktury použité na DagsHubu.

Struktura projektu a použitých souborů je naznačena v následujícím stromu:

/
- `bci-classifier` – Použitý klasifikátor Romana Kalivody
  - `bciclassifier` – Pythonovské utility využívané v klasifikátoru i v práci pro načtení dat
- `bciclassifier_data` – Složka datových souborů
  - `audio` – složka všech audio stimulů
  - `audiovisual` – složka všech audiovizuálních stimulů
  - `gen` – složka vygenerovaných souborů
  - `visual` – složka všech vizuálních stimulů
- `test` – Složka obsahuje testovací utility pro ověření správnosti řešení
- `bciclassifier-demo.ipynb` – Jupyter Notebook klasifikátoru Romana Kalivody
- `Combine channels.ipynb` – Jupyter Notebook pro generování dat s kombinovanými kanály
- `GAN.ipynb` – Jupyter Notebook pro generování dat pomocí GAN sítě
- `Noise generation.ipynb` – Jupyter Notebook pro generování dat s přidaným šumem
- `Time move MNE.ipynb` – Jupyter Notebook pro generování dat s časovým posunem

Struktura je členěna do tří hlavních celků:

- datové soubory
- pomocné utility
- Jupyter notebooky

Složka `bciclassifier` obsahuje veškeré materiály od Romana Kalivody. V rámci složky najdeme Jupyter Notebook samotného klasifikátoru a také složku se shodným názvem. Tato složka obsahuje soubory `constants.py` a `data_manager.py`, které slouží pro definování konstant a načítání datových souborů. Tyto soubory jsou využívány v rámci bakalářské práce (načítání souborů).

Složka `bciclassifier\_data` obsahuje veškerá data pro trénování klasifikátoru. V rámci složky jsou datové soubory rozděleny dle typu stimulů do jednotlivých složek (audio, audiovisual, visual). V rámci jednotlivých složek jsou soubory pro jednotlivé subjekty. Typ dat (test, train), číslo subjektu (1 až 16) a typ stimulu (A-audio, AV-audiovisual, V-visual) můžeme vyčíst z názvu datového souboru (například: *s1_A_button.dat*)

Soubory `Combine channels.ipynb`, `GAN.ipynb`, `Time move MNE.ipynb` a `Noise generation.ipynb`

5.4.2 Technologie experimentu

V rámci experimentu bylo na výběr mnoho technologií (viz výše). Na základě rozhodnutí (zmíněných níže) byly vybrány tyto technologie:

- programovací jazyk Python 3
- Jupyter Notebook
- MNE Tools
- DagsHub.com - MLFlow

Výběr programovacího jazyka Python byl ovlivněn zadáním bakalářské práce. Po vzoru práce Romana Kalivody byl výkonný kód uložen v rámci *notebooků*. Pro běh *notebooků* byl zvolen **Jupyter Notebook**. Při seznamování se s projektem byl většinou využíván **Google Colab**. Běh experimentu v tomto prostředí však byl negativně ovlivněn dostupnou RAM paměti v rámci experimentu. Experiment byl často během běhu ukončen pro vyčerpání veškeré RAM paměti. Z tohoto důvody byl běh přesunut do prostředí **Jupyter Notebook**, který umožňoval využití veškerých dostupných prostředků. Na

doporučení vedoucího bakalářské práce byly výsledky měření zaznamenány a sbírány centrálně v rámci nástroje **MLFlow**. Pro tento účel byl zřízen účet v rámci služby *DagsHub.com*, která nabízí v rámci cloudového prostředí i službu *MLFlow*.

5.5 Realizované rozšíření datového souboru

Pro generování dat bylo využito více způsobů vytváření a úprav vstupních dat. Mezi jednodušší metody generování můžeme zařadit například posun dat. Některé metody byly složitější, například GAN sítě. Při generování dat se postupovalo podle výše uvedené pipeline pro zpracování.

Načtená data byla ve formátu *.MAT*. Soubor obsahoval nejen naměřená data, ale i metadata. Mezi ně můžeme zařadit například informace o rozmístění sond, rozdělení epoch a datum a čas měření. Data naměřená jednotlivým subjektům byla rozdělena do několika setů. Pro každý subjekt byla vytvořena tato čtveřice (tlačítko, testovací data dvakrát, trénovací). Při generování jsem využíval dat trénovacích na vstupu generátoru. Vygenerovaná data byla následně dosamplována (doplněna) do tvaru (shape) vyžadovaného klasifikátorem pomocí datového souboru s1_A_train.dat.mat, který je tedy nutné mít v rámci příslušné složky. Nad nimi byly provedeny výše zmíněné úpravy. Následně byly datové soubory přidány a nahrazeny jako trénovací u jednotlivých subjektů. [7]

5.5.1 Časový posun

Výběr časových intervalů není možné určit podle emperického vzorce. Volí se vždy podle experimentu. V našem případě bylo rozhodnuto, vzhledem k rozestupu jednotlivých dat, pro posun o 0.1 a 0.2 sekundy. Posun byl v našem případě pomocí knihoven **mne.event.shift_time_events** a **mne.Evoked.shift_time**, která zajistila posun o výše stanovéné časové intervaly. [7]

Důležité zde bylo zachovat počet epoch a zajistit správný posun všech kanálů. Vhodné zvolení časového posunu bylo velmi důležité pro následné správné učení klasifikátoru. Při nevhodném zvolení časového intervalu pro posun by mohlo nastat přeučení klasifikátoru pro nás nevhodným způsobem.

5.5.2 Přidání šumu

Pro vygenerování šumu je potřeba kovarianční matice. V našem případě byla využita jednoduchá diagonální matice. Pro potřeby generování je nutné

zvolit hodnotu **SNR**. Hodnota SNR udává poměr síly signálu vůči síle šumu v pozadí. Pokud hodnota SNR přesáhne poměr 1.0/ 1.0 označuje situaci [11], kdy je více signálu než šumu. Hodnotu jsem volil experimentálním způsobem s výběr poměru s nejlepšími výsledky. [7]

```
info = data2.info

cov = mne.cov.make_ad_hoc_cov(info)
snr = 15.
cov['data'] *= (20. / snr) ** 2
mne.simulation.add_noise(data, cov=cov,random_state=0)
```

Ukázka kódu 2: Generování šumu

Pro potřeby generování šumu jsem využil knihovny `MNE`. Pomocí funkce *mne.cov.make_ad_hoc_cov()* jsem dogeneroval kovarianční matici. Pro generátor bylo po experimentálním vyzkoušení několika hodnot zvoleno SNR = 20 / 15, se kterým bylo dosaženo na vstupních datech nejlepších výsledků Následně je kovarianční matice naškálovaná (umocněná 2), aby se dosáhlo cíleného SNR. Při používání funkce *mne.cov.make_ad_hoc_cov* se vyskytli tzv. Warningy (bylo otevřeno i několik ticketů na Gitlab, nebylo však aktivně řešeno, Warningy jsou součást i referenčního kódu Jupyter Notebooku z dokumentace). Ukázku implementace v bakalářské práci najdete v snippetu 2. [9] [10]

5.5.3 Kombinace předchozích metod

Další možností, jak dosáhnout rozšíření datového souboru, je kombinace předchozích metod. Budeme-li kombinovat předchozí metody, dosáhneme jak robustnosti, tak vylepšení v klasifikaci.

V rámci práce byl kombinován posun a vygenerovaný šum. Pro spojení signálu bylo využito metody `mne.concatenate_epochs`. Výsledný soubor byl uložen na úložiště. Ukázkovou implementaci najdete ve snipptetu 3. [1]

```
data_combined =  mne.concatenate_epochs([noise,move])
```

Ukázka kódu 3: Kombinace kanálů

5.5.4 GAN sítě

Pro potřeby realizace GAN sítí bylo prozkoumáno několik již realizovaných publikovaných řešení.

V rámci článku *EEG Signal Reconstruction Using a Generative Adversarial Network With Wasserstein Distance and Temporal-Spatial-Frequency Loss* [22] jsou uváděny různé postupy pro rekonstrukci signálu. Jedním z nich je realizace pomocí GAN sítě s využitím Wassersteinovy vzdálenosti.

Dle *EEG data augmentation for emotion recognition using a conditional Wasserstein Gan* [23]

Ná základě článku *EEG data augmentation for emotion recognition using a conditional Wasserstein Gan* [23] bylo rozhodnuto o replikování popsaného experimentu podobným způsobem. Volba rekonstrukce GAN sítě dle tohoto článku byla zvolena z důvodu menších nároků na realizované řešení. Zmiňovaná metoda je *state of the art* mezi zpracováním EEG signálů. Architektura generátoru a diskriminátoru vychází také z tohoto článku (pro potřebu vstupních dat jsem upravil počet vrstev - viz 5.3.3). V rámci bakalářské práce byla tato změna provedena pro lepší přizpůsobení experimentu vstupnímu datovému souboru.

Počet vnitřních vrstev byl optimalizován dle kapitoly 5.3.3. Batch size byla stejně jako v referenčním článku zvolena experimentálně na 32 (čtvrtinové oproti 128 uvedených jako minimum v referenčním článku). Na rozdíl od článku byl zvolen `rmsprop` optimizer z důvodu dřívějších zkušeností.

Pro vytvoření generátoru jsem využil knihovny `Keras`. Díky ní jsem vytvořil model pro generátor. Velikost vstupních dat byla odvozena od velikosti samplovaných vstupních dat. Dále jsou ve 3 skupinách **Dense** vrstva a aplikována **LeakyReLu** funkce a **Batch normalization** (viz. obrázek 8.1). V každé skupině se zvyšuje velikost Dense vrstvy. Na závěr je změněna velikost modelu pomocí metody reshape.

Model diskriminátoru (viz. obrázek 8.2) jsem se koncepčně snažil stavět podobně referenčnímu článku. V rámci modelu se střídají vrstvy opět LeakyReLu a Dense. Aktivace probíhá pomocí funkcí sigmoid.

6 Testování

Pro software zpracovávající EEG data je důležité, aby byl spolehlivý a robustní. Tyto vlastnosti je třeba udržet a kontrolovat. K tomu slouží různé testovací metody a postupy. Využíváme-li při práci různé aplikace, máme na ně různé technické požadavky.

6.1 Jednotkové testování

Možnost využití jednotkových testů pro potřeby této práce je omezená. Byl vytvořen test pro ověření funkce GAN sítě na bázi vstupní sinusoidy.

6.2 Manuální testování

Testování některých komponent bylo obbtížné. Typickým příkladem těchto částí je šum. Proto tyto komponenty byly otestovány manuálně, a to porovnáním se vstupními daty.

6.3 Testování aplikace

Správnost fungování metod pro rozšiřování datasetu byla ověřena pomocí jednotkových testů. Ve všech případech se využívalo vygenerováné sinusoidy, která reprezentovala EEG signál. Sinusoida byla použita na vstupu jako referenční zdroj signálu, vůči němuž byly testovány jednotlivé metody pro rozšíření datového souboru.

6.4 Testovací scénáře

Pro ověření správnosti realizovaných metod bylo nutné otestovat správnou funkčnost jednotlivých úprav. Byly vytvořeny testovací scénář a byl vytvořen referenční vstup, který byl využit pro zvolené testovací scénáře. Testování bylo prováděno také manuálně.

6.4.1 TC 1 - Posun

Na načtená vstupní data byly aplikovány metody, které slouží pro augmentaci dat. Pro porovnání změn byly realizovány vizualizace pro jednotlivé

typy dat. Na základě těchto výsledků bylo porovnáno dosažených změn ve vstupních datech.

6.4.2 TC 2 - Šum

Vstupní data byla pozměněna pomocí aplikovaných metod, které slouží pro augmentaci dat. Testování funkčnosti této metody bylo obtížné, proto bylo opět zvoleno řešení stejné jako u posunu. Pro porovnání změn byly realizovány vizualizace. Na základě těchto výsledků bylo porovnáno dosažených změn ve vstupních datech a porovnány dosažené výsledky se vstupem.

6.4.3 TC 3 - GAN síť

U GAN sítě bylo ověřování správné funkčnosti sítě obtížnější z důvodu používání neuronové sítě (princip nejistoty). Ověření probíhalo opět vůči předgenerované sinusoidě. Řešení bylo provedeno na základě článku [12]. Řešení bylo přizpůsobeno reálnému experimentu. Bylo nutné vygenerovat testovací i trénovací sinusoidu. Následné bylo zkontrolováno, zda se vygenerovaný signál blíží sinusovému charakteru.

6.4.4 TC 4 - kombinace

Pomocí pytestu bylo v rámci Jupyter Notebooku otestována funkce, která se stará o kombinování dat z jednoho datového souboru a druhého datového souboru.

6.5 Zhodnocení testování

Realizace některých z těchto scénářů byla jednodušší (posun), u některých bylo složitější testovat, zda plní stanovený účel, protože na výstupu se projevoval náhodný jev již z povahy prováděného experimentu.

Veškeré připravené metody rozšíření se úspěšně podařilo otestovat. Realizované a implementované metody pro rozšíření jsou ověřeny a fungují dle předdefinovaných a stanovených principů.

7 Zhodnocení výsledků

V rámci této práce bylo realizováno více druhů experimentů. Jejich výsledky byly závislé jak na zvolené metodě, tak na velikosti datasetu. V rámci experimentu hrála významnou roli volba použité metody. Experiment probíhal vždy ve třech měřeních, ze kterých byly spočteny průměrné hodnoty, průměrné odchylky a maximální naměřené hodnoty po vzoru jiných prací.

Experimenty byly založeny na nastudovaných metodách augmentace dat a na základě vědeckých článků. Bylo vyzkoušeno několik odlišných způsobů a byly porovnány jejich výsledky. Pro potřeby práce byly voleny poměry generování na základě doporučených hodnot nebo dle vzorových experimentů.

7.1 Typy realizovaných experimentů

V rámci této práce bylo řešeno několik připravených experimentů. Každý experiment používal jednu z výše uvedených augmentačních metod (šum, posun, GAN).

V rámci referenčního článku [23] bylo generováno 200, 500, 1000, 3000, 5000, 10000, 15000 a 20000 epoch. Rozhodl jsem se postupovat obdobným způsobem. Vzhledem k časové náročnosti měření výsledků byly zvoleny obdobné hodnoty z nižšího oboru výše uvedeného spektra (200 - ekvivalent přidání 30% v tamním experimentu, 500 - ekvivalent 70 %, 1000 - 150 %, ...).

Pracoval jsem s následujícími datovými soubory:

- Originální data - originální nepozměněná data, tj. původní experiment
- Kombinované - kombinace šum + posun
- Přidání 50% dat - přidání k trénovacím datům poloviční množství epoch navíc
- Přidání 100% dat - přidání k trénovacím datům stejné množství epoch navíc
- Přidání 150% dat - přidání k trénovacím datům jeden a půl násobku množství epoch navíc

7.2 Parametry a nároky na měření

Jelikož se jedná o úlohu z oblasti strojového učení, je vhodné pro výpočty využít pracovní stanici s grafickou kartou, která umožňuje akceleraci výpočtu. Dále je nutné mít dostatek úložného prostoru (například při využití webového rozhraní Google Colab s úložištěm Google Drive). Aktuální rozšířený dataset zabírá 5,9 GB úložného prostoru. Vzhledem k velikosti datasetu je také nutné mít dostatečnou velikost RAM (při běhu v rámci Google Colab s vyhrazenými 12 GB RAM nebylo možné spouštět experiment s rozšířeným datasetem - nestačila při běhu RAM).

Pro rekapitulaci experimentu je nutné mít nainstalovaný `Python 3`, dále balíčkovací manažeř `pip`, běhové prostředí `Jupyter Notebook` spolu s `Anaconda Navigator`.

Z časového hlediska bylo generování a běh *bci-classificatoru* zhruba stejně dlouhé - v průměru 67 minut. Se vzrůstající délkou vygenerovaných dat vzrostla délka běhu *bci-classificatoru* na 95 minut (s výchylkou 160 minut) při největší velikosti vstupního datasetu.

7.3 Dosažené výsledky

Při měření natrénovaných modelů na trénovacím a testovacím datovém souboru bylo dosaženo těchto hodnot:

Experiment	Trénovací accuracy	Testovací accuracy
Originální data	0,96282	0,83612
Kombinované	0,96501	0,83336
Přidání 50% dat	0,97980	0,84208
Přidání 100% dat	0,98661	0,86721
Přidání 150% dat	0,99530	0,86859

Tabulka 7.1: Tabulka porovnání natrénovaných modelů na testovacích a trénovacích datech

V rámci experimentu byl naučen model GAN sítě pro potřeby generování dat. V tabulce 7.1 jsou zaneseny hodnoty výsledné accuracy pro naučený model spuštěný na trénovací množině dat (Trénovací accuracy) a také spuštěný na testovací množině dat (Testovací accuracy). Výsledky potvrzují předpoklad vzrůstu accuracy při zvolení augmentačních technik, tedy smysl použití vybraných technik.

Výsledky měření na testovacích datech[1]:

Experiment	Průměrná accuracy	Max accuracy	Průměrná odchylka
Originální data	0,83612	0,84352	0,00372
Kombinované	0,84336	0,85021	0,00184
Přidání 50% dat	0,84208	0,85071	0,00230
Přidání 100% dat	0,86721	0,86746	0,00024
Přidání 150% dat	0,87209	0,87562	0,00353

Tabulka 7.2: Tabulka naměřených hodnot na testovacích datech

Měření probíhalo pro celkovou accuracy (přesnost) společně pro audio, audiovizuální a vizuální stimuly, ale i pro accuracy pouze na audiovizuálních stimulech simulujících počáteční demenci. V rámci tabulky 7.2 jsou zaneseny pouze hodnoty pro celkovou accuracy (audiovizuální stimul je zanesen v následující tabulce).

Výrazné zlepšení výsledků při trénování je možné nalézt u přidávání pomocí GAN sítě. Zlepšení nastalo nejen v oblasti accuracy, ale také v oblasti odchylky měření. Zajímavým jevem u GAN sítí bylo zmenšování průměrné odchylky v závislosti na zvyšující se velikosti datového souboru. Nejvyšší hodnoty (0,87562) bylo dosaženo při měření 150% přidaných dat. Naopak největší odchylky (0,00372) bylo dosaženo při měření originálních dat.

Výsledky měření na testovacích audiovizuálních datech[2]:

Experiment	Průměrná accuracy	Max accuracy	Průměrná odchylka
Originální data	0,59971	0,61221	0,00866
Kombinované	0,60660	0,60880	0,00227
Přidání 50% dat	0.60864	0.61164	0,00448
Přidání 100% dat	0,55126	0,56470	0,01344
Přidání 150% dat	0,54293	0,54621	0,00328

Tabulka 7.3: Tabulka naměřených audiovizuálních testovacích hodnot

Jak již bylo zmíněno, měření probíhalo i pro samostatný audiovizuální stimul. Výsledky měření jsou zaneseny v rámci tabulky 7.3. Tabulka popisuje průměr hodnot, maximální výchylku i průměrnou odchylku.

V rámci naměřených hodnot je možné vidět zlepšení výsledků kombinovaných metod oproti GAN sítím. Velikost rozptylu se měnila v závislosti na velikostí dogenerovaných dat. Pozitivních výsledků bylo dosaženo při použití

[1]pro všechny typy stimulů - audio, audiovisual, visual

[2]jen pro **AUDIOVIZUÁLNÍ** stimuly

kombinovaných metod. Oproti předchozí analýze vzrostla celková accuracy a zároveň více vzrostla i odchylka.

Výsledky měření metriky **precision** na celkovém datovém souboru:

Experiment	Průměrná precision	Max precision	Průměrná odchylka
Originální data	0.62119	0,62611	0,03423
Kombinované	0,60926	0,61056	0,00666
Přidání 50% dat	0.62233	0.62691	0,04482
Přidání 100% dat	0,61483	0,64566	0,03083
Přidání 150% dat	0,62288	0,62711	0,00423

Tabulka 7.4: Tabulka hodnot precision naměřených na testovacích hodnot

Z pohledu metriky precision nastal propad u kombinovaných metod spolu s nárůstem průměrné odchylky. Hodnoty jsou zaneseny v rámci tabulky 7.4. Zlepšení se opakovalo u GAN sítí s maximálním zlepšením u přidávání 150% generovaných dat. Maximální naměřenou hodnotu najdeme u přidávání 100% dat (0,62288). Celkové zlepšení proti originálním datům je o 0,2%. Měření metriky precision nebylo součástí referenčního článku, proto nemůže být dále porovnáno dosažení stanovených hodnot.

Celkové zlepšení u sdružených audio, audiovizuálních a vizuálních dat je o 3,6%. Zlepšení v rámci samotného audiovizuálního datasetu nenastalo. Oproti původnímu experimentu je zlepšení méně výrazné. V rámci referenční práce bylo na podobném datasetu (DEAP - 15 subjektů, 32 experimentů, 4 vrstvy generátoru a diskrimintáoru) dosaženo pomocí WGAN zlepšení z 42,7% na 47,6%, tedy zlepšení o 4,9%. Hodnoty 42,7% bylo dosaženo na datasetu s 0 přidanými daty - ekvivalent originálu v této bakalářské práci. Pracovalo se také s 2000 přidanými daty (výsledek 47,6%). Takový rozsah však nemohl být v bakalářské práci z časových důvodů realizován.

Při porovnání s menším počtem přidaných dat vychází srovnání lépe. Při porovnání zlepšení z 42,7% (0 přidaných data - ekvivalent originálu) na 45,8% (500 přidaných dat oproti tamnímu datovému souboru s 480 vzorky odpovídá přidání zhruba o 105%, v bakalářské práci je srovnáváno s přidáváním o 100%). V rámci referenčního článku bylo dosaženo zlepšení o 3,1%. V rámci této bakalářské práce bylo dosaženo zlepšení také o 3,1% při porovnání originálu (dosaženo hodnoty 83,6%) a 100% (dosaženo hodnoty 86,7%) přidávání dat.

8 Závěr

Při přípravě bakalářské práce byla nejprve prostudována problematika záznamu činnosti mozku a zpracování signálu elektroencefalografu - zvláště reakce na evokované potenciály. Na základě prostudování metod rozšíření datové sady byly vybrány metody, které se jevily pro tuto bakalářskou práci nejvhodnější (posun, šum, kombinace posun a šum, GAN sítě). Pomocí těchto technik byly vytvořeny další datové soubory rozšiřující původní datový soubor. Využito bylo i výsledků a zkušeností zveřejněných v odborné literatuře. Při práci s GAN sítěmi bylo čerpáno především z článku *Data augmentation for enhancing EEG-based emotion recognition with deep generative models*. Původní i rozšířená datová sada byly následně uplatněny v klasifikační úloze. Použit byl klasifikátor Romana Kalivody. Při následné implementaci v jazyce Python bylo nutné změnit běhové prostředí (Google Colab -> Jupyter Notebook) z důvodu nedostatečné kapacity cloudového řešení.

Původní data i syntetická data byla porovnána za využití vybraných metrik (accuracy, precision). Bylo prokázáno zlepšení, nejlepší výsledky vycházely v rámci syntetických dat vygenerovaných v rámci GAN sítí. Procentuální zlepšení při této metodě zhruba odpovídalo zlepšení, jakého bylo dosaženo v rámci referenčního odborného článku (3,1%).

Bakalářská práce splnila veškeré body zadání, výsledkem je rozšířený datový soubor, jehož funkčnost a robustnost byla ověřena několika metrikami (accuracy, precision)

Seznam obrázků

Seznam ukázek kódu

Seznam použitých zkratek

A Audio stimul

AV Audiovizuální stimul

V Vizuální stimul

ML Machine Learning

EEG Elektroencefalograf

GAN Generative adversial network

SNR Podíl signálu a šumu

RAM RAM paměť (Random access memory)

Literatura

[1] *MNE Tools mne.concatenate_epochs - mne 1.0.3 documentation* [online]. [cit. 2022/06/22]. Dostupné z: https://mne.tools/stable/generated/mne.concatenate_epochs.html.

[2] *Neuroscience For Kids - 10-20 System* [online]. [cit. 2022/02/05]. Dostupné z: https://faculty.washington.edu/chudler/1020.html.

[3] Fair principles, Jan 2022. Dostupné z: https://www.go-fair.org/fair-principles/.

[4] *153ZODH / 5. cvičení* [online]. Dostupné z: https://geo.fsv.cvut.cz/gwiki/153ZODH_/_5._cvi%C4%8Den%C3%AD.

[5] Postup zapojení EEG V Brmlabu. Dostupné z: https://brmlab.cz/project/brain_hacking/eeg.

[6] *Co je hluboké učení?* [online]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/what-is-deep-learning/.

[7] *MNE Tools Epochs* [online]. [cit. 2022/05/01]. Dostupné z: https://mne.tools/stable/generated/mne.Epochs.html.

[8] EEG checkerboard pattern of bruxism, Sep 1999. Dostupné z: https://n.neurology.org/content/53/4/669.

[9] *Mne.simulation.add_noise mne.simulation.add_noise - MNE 1.1.dev0 documentation* [online]. [cit. 2022/05/03]. Dostupné z: https://mne.tools/dev/generated/mne.simulation.add_noise.html.

[10] *DICS for power mapping - MNE 0.20.7 documentation* [online]. [cit. 2022/03/01]. Dostupné z: https://mne.tools/0.20/auto_tutorials/simulation/plot_dics.html.

[11] *Signal to noise ratio formula* [online]. Mar 2022. [cit. 2022/04/05]. Dostupné z: https://www.geeksforgeeks.org/signal-to-noise-ratio-formula/.

[12] *1D GAN for Sine Wave function* [online]. [cit. 2022/06/21]. Dostupné z: https://saifgazali.medium.com/1d-gan-for-sine-wave-function-f7ea81e99c37.

[13] Azure. *Neuronová síť*. Dostupné z: https://studuj.digital/wp-content/uploads/2020/09/image-9.png.

[14] Berger, I. – Cassuto, H. The effect of environmental distractors incorporation into a CPT on sustained attention and ADHD diagnosis among adolescents. *Journal of neuroscience methods*. 11 2013, 222. doi: 10.1016/j.jneumeth.2013.10.012.

[15] Fink, A. et al. Creative Ways to Well-being: Reappraisal Inventiveness in the Context of Anger Evoking Situations. *Cognitive Affective Behavioral Neuroscience*. 11 2016, 17. doi: 10.3758/s13415-016-0465-9.

[16] Gonfalonieri, A. *Data augmentation for Brain-Computer interface* [online]. Towards Data Science, Mar 2021. [cit. 2022/06/01]. Dostupné z: `https://towardsdatascience.com/data-augmentation-for-brain-computer-interface-35862c9beb40`.

[17] Hrazdira, I. – Mornstein, V. *Lékařská biofyzika a přístrojová technika*. Neptun, 1. edition, 2001. ISBN 80-902896-1-4.

[18] Kemp, B. e. a. *Standard texts and polarity rules* [online]. 2003. [cit. 15.03.2022]. Dostupné z: `https://www.edfplus.info/specs/edftexts.html`.

[19] Koupilová, Z. Souhrnný Sborník Veletrhu nápadů učitelů fyziky. Dostupné z: `https://vnuf.cz/sbornik/prispevky/16-01-Balek.html`.

[20] Liu, Y. – Chen, Z. – Guo, F. *Big data market segmentation*. Createspace Independent Publishing Platform, June 2013.

[21] Luck, S. J. An Introduction to the Event-Related Potential Technique. 2005.

[22] Luo, T.-j. et al. EEG Signal Reconstruction Using a Generative Adversarial Network With Wasserstein Distance and Temporal-Spatial-Frequency Loss. *Frontiers in Neuroinformatics*. 2020, 14. ISSN 1662-5196. doi: 10.3389/fninf.2020.00015. Dostupné z: `https://www.frontiersin.org/article/10.3389/fninf.2020.00015`.

[23] Luo, Y. – Lu, B.-L. EEG data augmentation for emotion recognition using a conditional Wasserstein Gan. *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2018. doi: 10.1109/embc.2018.8512865.

[24] Luo, Y. et al. *Data augmentation for enhancing EEG-based emotion recognition with deep generative models* [online]. Jun 2020. [cit. 2022/05/23]. Dostupné z: `https://arxiv.org/abs/2006.05331`.

[25] Minaee, S. 20 popular machine learning metrics. part 1: Classification amp; Regression Evaluation Metrics, Oct 2019. Dostupné z: https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-\regression-evaluation-metrics-1ca3e282a2ce.

[26] Pokorný, d. M. J. Page 1/5 April 24, 2020, Kladno - fbmi.cvut.cz. Dostupné z: https://web.archive.org/web/20140628160804/https://www.fbmi.cvut.cz/files/nodes/657/public/EEG.pdf.

[27] Prof. Ing. Václav Matoušek, C. [online]. KIV ZČU, 2016. [cit. 2022/03/01]. Umělá inteligence a rozpoznávání. Dostupné z: https://www.kiv.zcu.cz/studies/predmety/uir/.

[28] Sanei, S. – Chambers, J. A. *1.3 Action potentials*. John Wiley amp; Sons, 2009. Dostupné z: https://www.amazon.ca/gp/product/0470025816? ISBN 978-0470025819.

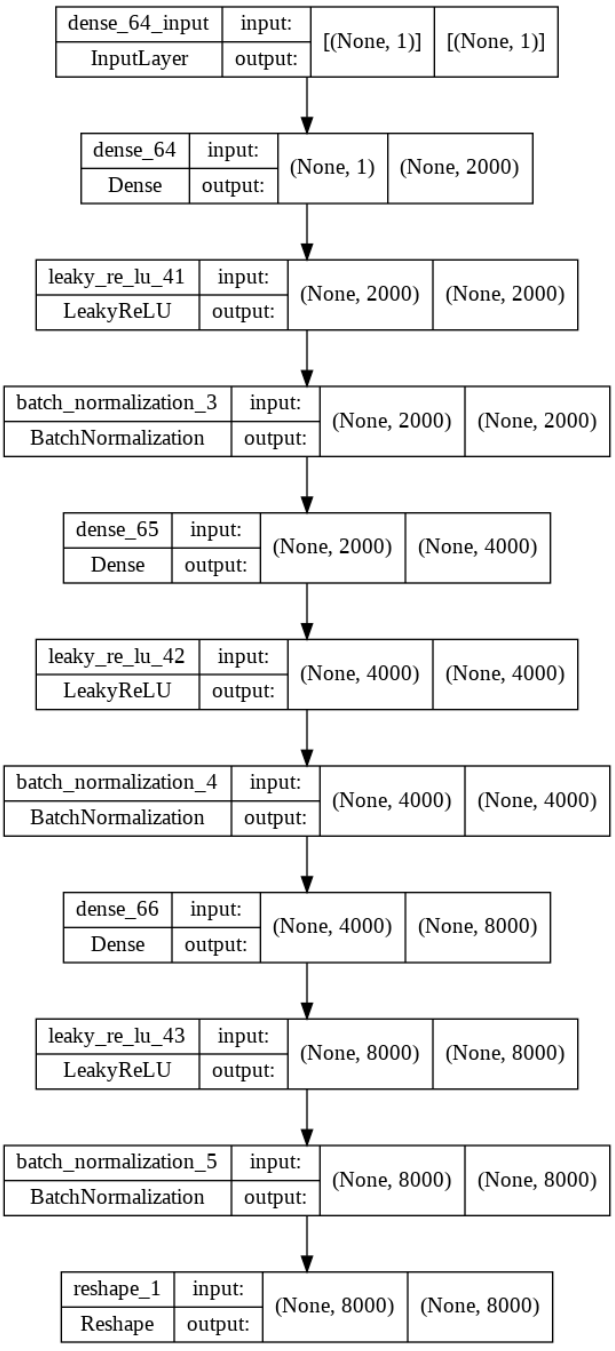
[29] *SMOTE for Imbalanced Classification with PythonD* [online]. Machine Learning Mastery, 2020. [cit. 15.06.2022]. Dostupné z: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/.

[30] staff, S. X. The brain learns completely differently than we've assumed since the 20th Century, Mar 2018. Dostupné z: https://medicalxpress.com/news/2018-03-brain-differently-weve-assumed-20th.html.

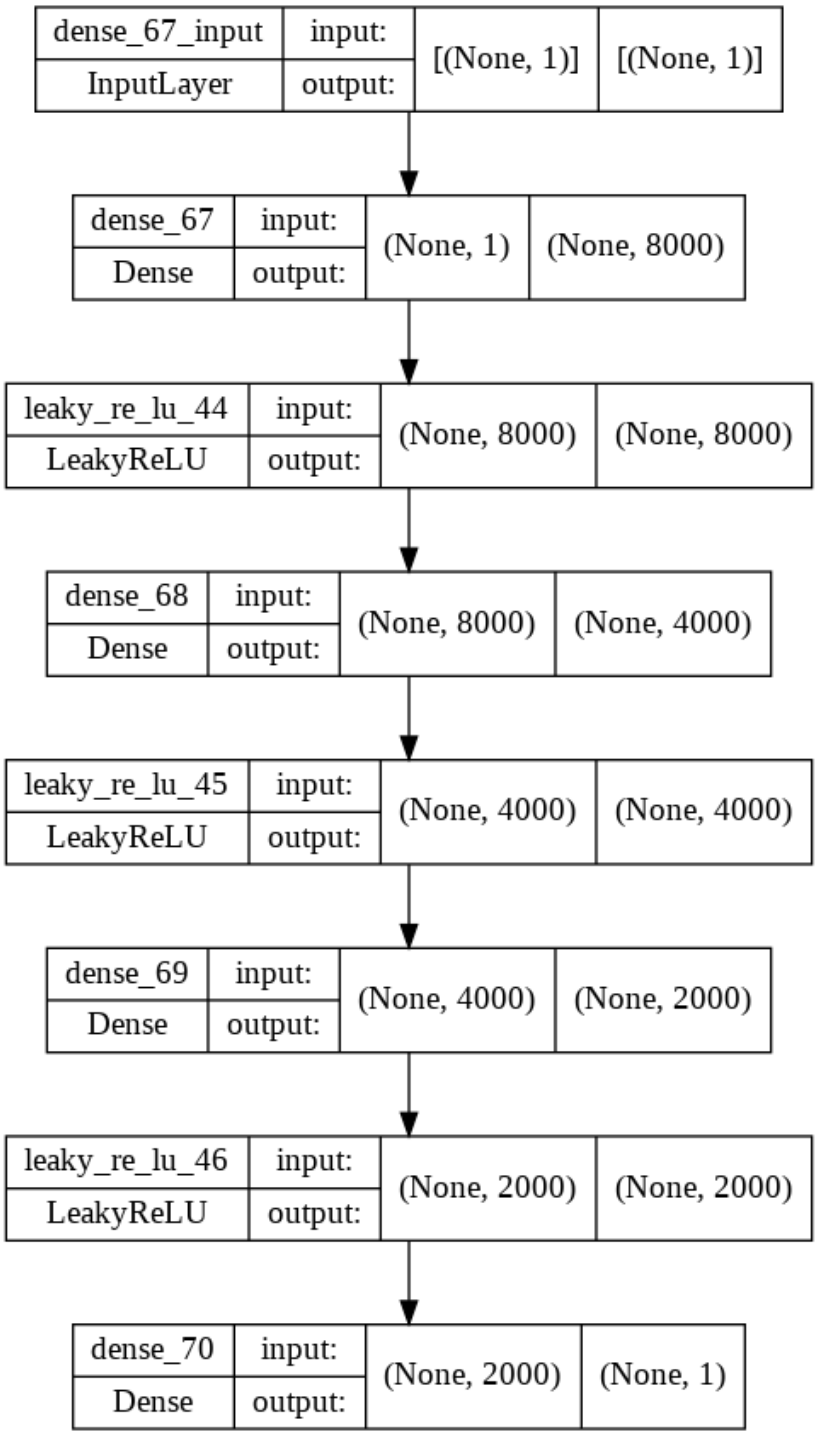
[31] Tavakolian, M. – Cruces, C. – Hadid, A. Learning to Detect Genuine versus Posed Pain from Facial Expressions using Residual Generative Adversarial Networks, 05 2019.

[32] Vařeka, L. Evaluation of convolutional neural networks using a large multi-subject P300 dataset. *Biomedical Signal Processing and Control*. Apr 2020, 58, s. 101837. ISSN 1746-8094. doi: 10.1016/j.bspc.2019.101837. Dostupné z: http://dx.doi.org/10.1016/j.bspc.2019.101837.

Přílohy



Obrázek 8.1: Model generátoru, zdroj: realizace práce



Obrázek 8.2: Model diskriminátoru, zdroj: realizace práce

Uživatelská příručka

V této příručce je popsáno jak provozovat bakalářskou práci. Je uveden popis technologií potřebných pro běh práce.

Technologie

Pro hladký běh programu je předpokládána instalace následujících technologií:

- `Anaconda3` - distribuce balíčků Python, R pro vědecké výpočty
- `Jupyter Notebook` - webové běhové prostředí pro Python
- `pip` - Balíčkovací manager
- `Python` 3 - programovací jazyk

Pro běh jsou potřeba také tyto balíčky:

- `DataManager` - knihovna od studenta Romana Kalivody - pro načítání dat
- `numpy` - knihovna v Pythonu poskytující infrastrukturu pro práci s vektory, maticemi a obecně vícerozměrnými poli
- `MNE Tools (mne.Epochs, mne.channels, mne.cov)` - knihovna v Pythonu umožňující práci s EEG, MEG, sEEG, ... daty na úrovni prozkoumávání, analýzu a vizualizaci
- `scipy` - poskytuje v Pythonu algoritmy například pro optimalizace, integrace, interpolace, albegraické výpočty, diferenciální výpočty, statistiku
- `matplotlib` - knihovna umožňující statické, anomované a interaktivní vizualizace
- `Tensorflow` - knihovna pro strojové učení a umělou inteligenci
- `tqdm` - progress-bar pro vizualizaci průběhu činnosti
- `plot_utils` - generátor 2D grafik na základě dat

- `pytest` - knihovna pro testování v Pythonu

Možnou alternativou pro běh je `Google Colab`, který běží kompletně v cloudovém prostředí (na externí výpočetní infrastruktuře připravené pro běh specializovaných aplikací). Touto technologií se nechají nahradit veškeré technologie kromě `MNE Tools` (také běží v cloudovém prostředí - nutná registrace)

Sestavení programu

Program by mělo být možné spouštět na všech operačních systémech (pip, python3 i anaconda jsou dostupné pro Linux, MacOS i Windows). Vývoj však probíhal a byl testován pouze na Windows. Program běží ve webovém prohlížeči (Jupyter Notebook i Google Colab) a využívá se služeb `pip` pro instalaci potřebných závislostí. Instalace závislostí probíhá v rámci Jupyter Notebooků. Není tedy nutné mimo Jupyter Notebooky doinstalovávat žádné balíčky. Program využívá pro načtení dat, vizualizaci a ověření výsledků učení práce studenta Romana Kalivody.

Příprava složek

Pro potřeby generování je nutné dodržet následující schéma složek.

bciclassificator - výchozí klasifikátor

Přípravou pro běh `bciclassificatoru` je kontrola přesunutí vygenerovaných souborů (ze složky `gen` do příslušných složek dle typu stimulů) a následné smazání zbylých vygenerovaných dat ve složce `gen` - není možné spustit klasifikaci. Po otevření příslušného Juypter Notebooku se spustí klasifikátor.

GAN

Pro úspěšný chod programu je třeba vytvořit ve složce `bciclassifier_data` složku `gen`. Uvnitř této složky dále `A`, `AV` a `V` pro audio, audiovizuální a vizuální vygenerované datové soubory. Po dogenerování (generují se jako subject_17 a výše) je možné soubory ihned přesunout do příslušných složek audio, audiovisual a visual, které jsou na úrovni složky `gen`. Data se generují do oddělené složky pro možnost trénování dat na jedné množině datových souborů a souběžného dogenerovávání dat dalších.

Time move

Pro běh programu je potřeba vytvořit ve složce `bciclassifier_data` složku `gen-move`. Uvnitř této složky dále `A`, `AV` a `V` pro audio, audiovizuální a vizuální vygenerované datové soubory. Po dogenerování (generují se jako subject_17 a výše) je možné soubory ihned přesunout do příslušných složek audio, audiovisual a visual, které jsou na úrovni složky `gen`. Opět je nutné smazat složku gen před trénováním.

Combine channels

Pro běh programu je potřeba vytvořit ve složce `bciclassifier_data` složku `combine`. Uvnitř této složky dále `audio`, `audiovisual` a `visual` pro audio, audiovizuální a vizuální vygenerované datové soubory. Vygenerovaná data jsou umístěna ve výše zmíněných složkách. Složky je možné rovnou přetáhnout na úroveň složky bciclassifier_data. Složku opět před během klasifikátoru smažte.

Noise

Pro běh programu je potřeba vytvořit ve složce `bciclassifier_data` složku `noise`. Uvnitř této složky dále `A`, `AV` a `V` pro audio, audiovizuální a vizuální vygenerované datové soubory. Po dogenerování (generují se jako subject_17 a výše) je možné soubory ihned přesunout do příslušných složek audio, audiovisual a visual, které jsou na úrovni složky `gen`. Opět je nutné smazat složku noise před trénováním.

Spuštění programu

Pro úspěšné generování rozšířeného vstupního datového souboru je nutné dodržet podmínky stanovené výše (vytvoření složek, smazání složek). Dále si zkontrolujte odkaz na `dat_path` a odkazy na referenční soubor *s1_A_train.dat.mat* v rámci všech skriptů (odkaz na složku s datovými soubory). V případě `bciclassifier-demo.ipynb` si zkontrolujte nastavení MLFlow v dolní části Jupyter Notebooku.

Běh je možné spustit pro celý notebook v rámci Jupyter Notebooku (`Cell -> Run all`) i v rámci Google Colab (`Běh -> Spustit vše`).

9 Obsah elektronické přílohy

9.1 Adresářová struktura

- `/`
  - `Aplikace_a_knihovny`
    - `bci-classifier` – Použitý klasifikátor Romana Kalivody
    - `test` – Složka obsahuje testovací utility pro ověření správnosti řešení
    - `bciclassifier-demo.ipynb` – Jupyter Notebook klasifikátoru Romana Kalivody
    - `Combine channels.ipynb` – Jupyter Notebook pro generování dat s kombinovanými kanály
    - `GAN.ipynb` – Jupyter Notebook pro generování dat pomocí GAN sítě
    - `Noise generation.ipynb` – Jupyter Notebook pro generování dat s přidaným šumem
    - `Time move MNE.ipynb` – Jupyter Notebook pro generování dat s časovým posunem
  - `Text_prace`
    - `tex` – zdrojové soubory TEXvčetně přiložených souborů
    - `A19B0121P` – PDF soubor bakalářská práce
  - `readme.txt` – popis adresářové struktury