Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Identifikácia sekvenčných variantov v kontexte neistoty

Bakalárska práca

2021
Tomáš Janeta

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Identifikácia sekvenčných variantov v kontexte neistoty

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra aplikovanej informatiky
Školiteľ:	doc. Mgr. Tomáš Vinař, PhD.

Bratislava, 2021
Tomáš Janeta

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Tomáš Janeta
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Identifikácia sekvenčných variantov v kontexte neistoty
Identification of Sequence Variants with Uncertainty

Anotácia: Sekvenčné varianty, ako napríklad jednobázové polymorfizmy alebo krátke inzercie a delécie, sa typicky identifikujú prostredníctvom zarovnania sekvenačných čítaní ku referenčnej sekvencii. V rámci tohto procesu sa identifikuje konsenzus rozdielov voči referenčnej sekvencii. V kontexte nanopórového sekvenovania, kde čítania môžu obsahovať až 10% chýb, je toto veľmi ťažký problém. Cieľom práce je preskúmať nový prístup k identifikácii sekvenčných variantov, berúc do úvahy neistotu pri určovaní sekvencií v sekvenačných čítaniach.

Vedúci: doc. Mgr. Tomáš Vinař, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 13.10.2020

Dátum schválenia: 25.10.2020 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Týmto by som rád poďakoval svojmu školiteľovi Tomášovi Vinařovi za jeho čas, ochotu a trpezlivosť s mojím nesystematickým prístupom k práci.

Abstrakt

Hľadanie mutácií má široké využitie v medicíne a farmácii. V našej práci prezentujeme dva algoritmy na riešenie tohoto problému a porovnávame ich na rôznych typoch dát. Tiež skúmame vplyv použitia výstupu CTC vrstvy ako vstupné dáta pre tieto algoritmy.

Kľúčové slová: DNA, variant, nanopórové sekvenovanie

Abstract

Variant calling is widely used in modern medicine and pharmacy. We present two approaches to this problem and compare them on various data sets. Also we examine usage of output of CTC layer as input for these algorithms.

Keywords: DNA, variant, nanopore sequencing

Obsah

Úvod

Nositeľom genetickej informácie v prírode je DNA - deoxyribonukleová kyselina. Genetická infomácia je v nej uložená pomocou štyroch dusíkatých báz: adenínu, cytozínu, guanínu a tymínu. Zmena genetickej informácie sa nazýva mutácia. Mutácie môžu vznikať z rôznych dôvodov, napríklad v dôsledku vplyvu rádioaktívneho žiarenia alebo chemikálií.

Proces čítania DNA zo vzorky sa nazýva sekvenovanie. Nanopórové sekvenovanie je špecifická metóda sekvenovania DNA, pri ktorej vlákno DNA prechádza malým otvorom, pričom ovplyvňuje hodnotu elektrického prúdu v tomto otvore. Zo zmeny elektrického prúdu vieme určiť z akých báz sa vlákno skladá. Nanopórové sekvenovanie má mnoho výhod, medzi ktoré patria aj rýchlosť a nízka cena. Nevýhodou tejto metódy je, že dochádza k pomerne veľkým nepresnostiam.

Klasický postup pri hľadaní mutácií v DNA je taký, že najprv sa prečíta DNA zo vzorky, a potom sa v takto získaných dátach hľadajú varianty. V tejto práci sa pokúsime overiť hypotézu, či by dodatočná informácia ktorú získame z nanopórového sekvenovania mohla zvýšiť presnosť klasických algoritmov na hľadanie mutácií.

V prvej kapitole rozoberieme podrobne problém hľadania mutácií, jeho zložitosť a popíšeme momentálny stav výskumu v tejto oblasti. V druhej kapitole podrobne popíšeme algoritmy na hľadanie mutácií, ktoré sme implementovali, ako sme ich testovali a uvedieme výhody a nevýhody každého z algoritmov. V tretej kapitole presnejšie vysvetlíme princíp nanopórového sekvenovania aj to, ako ho chceme využiť v našej práci. V štvrtej kapitole vyhodnotíme vplyv nanopórového sekvenovania na výsledky implementovaých algoritmov. V piatej kapitole vysvetlíme implementačné detaily.

Kapitola 1

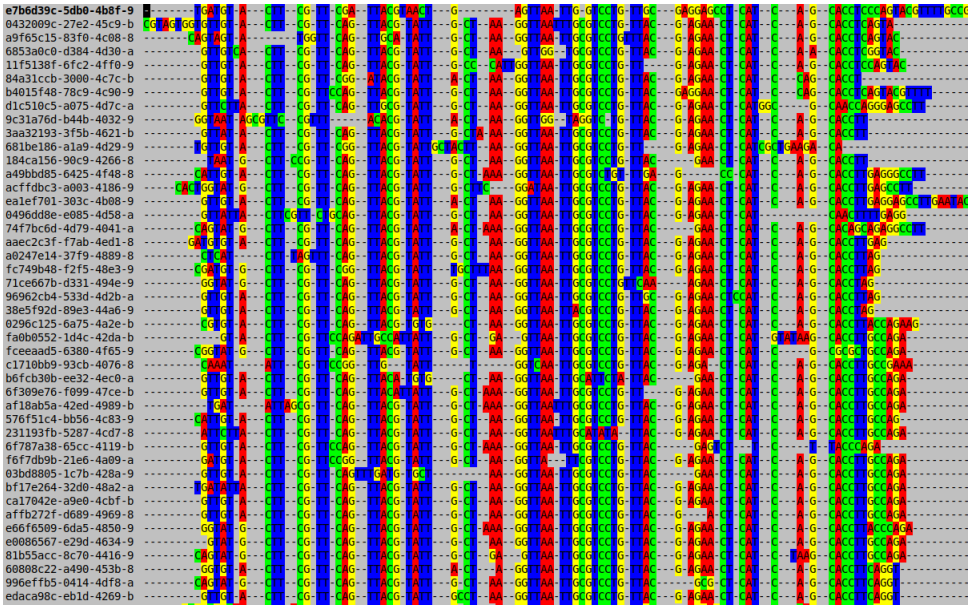
Problém hľadania variantov

1.1 Varianty v DNA

Najčastejší spôsob uchovania genetickej informácie v prírode je pomocou deoxyribonukleovej kyseliny (DNA). DNA je tvorená dvoma vláknami spletenými do tvaru dvojzávitnice. Každé vlákno obsahuje postupnosť dusíkatých báz anín, cytozín, guanín a tymín, v ďalšom texte označovaných ako A, C, G a T.
Dvojice báz A, T a C, G sú komplementárne. To znamená, že ak prvé vlákno má na istom mieste bázu A (resp. C) druhé tam bude mať T (resp. G) a naopak.
Teda na DNA sa môžeme pozerať ako na reťazec znakov nad abecedou $\{A, C, G, T\}$.

Proces zisťovania poradia báz v DNA sa nazýva *sekvenovanie*. V skutočnosti sa nečíta na jedenkrát celá DNA postupnosť, ale len jej časť. Aký dlhý úsek sa dá prečítať na jedenkrát závisí od konkrétnej metódy sekvenovania.
Krátke prečítané úseky DNA nazývame *čítania*. Čítania sa zvyčajne vzťahujú k dlhšej postupnosti DNA, ktorú nazývame *referencia*. Čítania popisujú istý krátky úsek DNA obsiahnutý v referencii.
Proces, na vstupe ktorého dostaneme 2 DNA postupnosti a snažíme sa zistiť, ktorému úseku v prvej postupnosti druhá postupnosť prináleží sa nazýva zarovnanie (angl. alignment). Cieľom zarovnávania je nájsť časť prvej postupnosti, ktorá sa čo najviac podobá na druhú.

Algoritmy zarovnávania sekvencií sa vo všeobecnosti delia na dve kategórie - *globálne*, ktoré sa snažia nájsť v prvej postupnosti globálne najlepší úsek tým, že ju celú prehľadajú, a lokálne, ktoré najprv nájdu krátke podpostupnosti vyskytujúce sa v obidvoch postupnostiach.
Existuje pomerne veľa nástrojov alebo softvérových balíčkov, ktoré možno použiť na zarovnávanie sekvencií. Príkladom je program minimap2 [14], ktorý sa používa na za-



Obr. 1.1: Čítania zarovnané pomocou nástroja muscle

rovnanie čítaní k referencii. Príklad použitia zarovnávacieho algoritmu je situácia, keď potrebujeme zistiť, ktorej časti referencie konkrétne čítania prináležia. V takom prípade sa dá využiť práve minimap2.

Na obrázku 1.1 možno pozorovať čítania pomocou nástroja muscle [7] a zobrazené pomocou softvéru seaview [9].

Medzi DNA postupnosťami jednotlivých jedincov v populácii je silná podobnosť, ale nie sú rovnaké. Medzi postupnosťami každých dvoch jedincov v populácii sú malé rozdiely. Práve v týchto rozdieloch sú zakódované špecifické vlastnosti jedinca, ako je farba očí, farba vlasov, odolnosť voči chorobám alebo náchylnosť na obezitu.
Variant je fenomén, keď sa niekoľko postupností DNA, ktoré by si mali byť veľmi podobné, líši len v malých detailoch. Príkladom je práve populácia jedincov toho istého druhu.

Existuje veľa druhov variantov, v našej práci sa sústredíme na tri najjednoduchšie: jednobázové polymorfizmy, inzercie a delécie:
SNP(angl. single nucleotid polymorphism) - jednobázový polymorfizmus je variant, ktorý vznikol tak, že sa jedna báza vymenila za inú. Napríklad ak sa z postupnosti **ACATGCA** stane **ACCTGCA**.
Delécia je variant, ktorý vznikol vymazaním jednej alebo viacerých susedných báz z pôvodnej postupnosti. Postupnosť **ACATGCA** zmenená na postupnosť **ACGCA** je príkladom delécie.
Inzercia je variant, ktorý vznikol vložením jednej alebo niekoľkých báz vedľa seba do

pôvodnej postupnosti. Postupnosť **ACATGCA** zmenená na postupnosť **ACATGCTCA** je príkladom inzercie.

1.2 Identifikácia variantov

Hľadanie variantov (angl. variant calling) má veľa využití, napríklad pri zisťovaní odolnosti baktérií na antibiotiká [4] alebo pri šľachtení špeciálnych druhov organizmov [2].

V práci navrhneme niekoľko prístupov na riešenie tohto problému, pričom sa sústredÍME na nasledujúcu modelovú situáciu. Vstupom bude referenčná postupnosť, zoznam sekvenovaných čítaní a pravdepodobnosť p. Žiadaným výstupom je zoznam variantov, pre ktoré je pravdepodobnosť, že sú to skutočne varianty, aspoň p (tým že sú skutočné myslíme, že nie sú spôsobené chybami merania, atď.).

Aj keď by sa mohlo zdať, že problém je jednoduchý, možné komplikácie si demonštrujeme na konkrétnom príklade. Uvažujme jednoduchý prístup, kde zarovnáme čítania k referencii a postupne porovnávame zarovnané dvojice pozícií v čítaní a v referencii. Vždy keď sa pozícia v čítaní nezhoduje s referenciou, vyhlásime takýto výskyt za variant. Na konci vyfiltrujeme iba varianty s dostatočne vysokou podporou čítaní. Vysvetlíme prečo tento prístup nefunguje vo všeobecnosti dobre.

Uvažujme že referenčná postupnosť je **ACATGCAAGCTT**, nech dva čítania obsahujú postupnosť **ACATGCAGCT** a začiatok čítania sa zarovná na začiatok referencie. V čítaniach sa nachádza delécia, pretože bol vymazaný znak **A**. Lenže zarovnávací program môže zarovnať pozíciu s písmenom **A** ktoré tam zostalo k dvom rôznym pozíciám v referencii a hore popísaný algoritmus nájde dva rôzne varianty.

Opísaný algoritmus sme aplikovali na vzorku obsahujúcu štyristo čítaní, z ktorých väčšina obsahovala približne 520 pozícií toho istého úseku. Vieme, že tento úsek by mal obsahovať niekoľko variantov a poznáme aj ich popis. Priemerná dĺžka čítaní nachádzajúcich sa vo vzorke bola 345 pozícií, vzorka pokrývala 340 pozícií, ktoré sa nachádzali vo viac ako 100 čítaniach. Algoritmus našiel hľadané varianty, ale aj veľa nesprávnych, ktoré nemal nájsť.
Zjavne tento algoritmus nie je vhodný na všeobecné použitie v praxi.

1.3 Nanopórové sekvenovanie

Nanopórové sekvenovanie je metóda sekvenovania DNA zo vzorky. V tenkej membráne sa vytvorí malý otvor, takzvaný nanopór. Na obidvoch stranách membrány sa nachádza elektrolyt. Vzniknutá sila ťahá vlákno DNA cez nanopór, ktorým prechádza elektrický prúd. Každá z báz A, C, G, T ovplyvňuje elektrický prúd v nanopóre špecifickým spôsobom. Zo zmeny prúdu vieme určiť bázu, ktorá prechádzala nanopórom [12].

Výhody tejto metódy sú rýchlosť, dlhé čítania a dostupnosť prvých dát už počas sekvenovania. Na druhej strane má metóda veľkú chybovosť, ktorá je spôsobená technickými problémami.
Hodnotu elektrického prúdu totiž neovplyvňuje len jedna báza, ktorá sa v ňom práve nachádza, ale aj k báz pred nanopórom a k báz za nanopórom, kde k je konštanta. Navyše vlákno sa v nanopóre nemusí hýbať rovnomernou rýchlosťou. Može zrýchliť, spomaliť, zastaviť či dokonca sa kúsok vrátiť späť. To výrazne sťažuje spracovanie surového signálu [20].

1.4 Momentálny stav výskumu v oblasti identifikácie variantov

Ako sme spomínali, hľadanie variantov má široké využitie. Vďaka tomu sa do výskumu v tomto smere investovalo množstvo prostriedkov a bolo vyvinutých veľa rôznorodých techník.
Jeden z najznámejších nástrojov používaných na hľadanie variantov sa volá GATK (angl. Genome Analysis Toolkit)[16]. Je to zbierka softérových nástrojov a algoritmov slúžiacich na analýzu zarovnaných čítaní s primárnym cieľom hľadania SNPov, inzercií a delécií. GATK sa sústreďuje hlavne na dáta pochádzajúce z ľudského genómu.
Existuje niekoľko softvérových nástrojov založených na metódach pravdepodobnosti a štatistiky.
Bcftools je časťou softwarového nástroja samtools slúžiaceho na manipuláciu, prevod medzi jednotlivými formátmi, filtrovanie a zobrazovanie zarovnaných čítaní.[6]
Freebayes je zovšeobecnením algoritmov GATK, Polybayes [21] a Bcftools. Freebayes funguje nasledovne: v čítaniach najprv nájde podozrivé pozície, ktoré sa nezhodujú s referenciou. Je veľká pravdepodobnosť, že tieto pozície sa nachádzajú v niektorom z variantov. Následne Freebayes vytvorí pre každú podozrivú pozíciu v každom čítaní úsek, ktorý ju obsahuje a je dostatočne veľký - takzvané okno. DNA postupnosť nachádzajúcu sa v okne potom porovná s referenciou.

Významný pokrok v tejto oblasti prinieslo aj strojové učenie. Medzi najznámejšie a najnovšie nástroje slúžiace na hľadanie variantov používajúce metódy z tejto oblasti patrí DeepVariant. DeepVariant je softwarový naštroj využívajúci metódy hlbokého učenia a neurónových sietí [17]. DeepVariant obsahuje konvolučnú neurónovú sieť so 6 vrstvami, z ktorých každá vypočíta jeden atribút výsledného variantu.
Ďalšie sú DeepSV[5] a NeoMutate[1].

Kapitola 2

Riešenie pomocou vymenovávania haplotypov

2.1 Algoritmus na hľadanie variantov

V našej práci sme sa inšpirovali nástrojom na hľadanie variantov Freebayes [8]. Freebayes funguje tak, že najprv nájde miesta, kde by sa potenciálne mohli nachádzať varianty. To sú miesta, kde sa referencia a čítania líšia. Freebayes ďalej nevyužíva dvojice navzájom zarovnaných pozícií, ale samotnú zarovnanú sekvenciu. Týmto spôsobom sa vyhne veľmi problematickému aspektu hľadania variantov - identické postupnosti môžu mať viacero zarovnaní.

Prístup, ktorý opíšeme, sa sčasti zakladá na algoritme použitom v nástroji Freebayes. Uvažujme vstup, ktorý obsahuje referenčnú postupnosť a zoznam čítaní. Najprv nájdeme všetky pozície, na ktorých sa čítania nezhodujú s referenciou. Takto vyfiltrované pozície majú veľký potenciál byť obsiahnuté v niektorom variante. V každom čítaní nájdeme všetky podozrivé pozície, ktoré obsahuje a rozšírime ich doprava a doľava o vopred určený počet pozícií. Všetky pozície, ktoré sa nachádzajú v takto vytvorenom intervale, spojíme do jedného reťazca. Na konci vyfiltrujeme reťazce, ktoré sa nezhodujú s referenciou a majú dostatočne vysokú podporu.

Tento algoritmus, ktorý sme implementovali, budeme pre jednoduchosť ďalej nazývať FB. FB algoritmus má tri vstupné parametre - a, r a p. Predpokladajme, že na niektorej pozícii sa nachádza variant, touto pozíciou prechádza n čítaní a tento variant podporuje m z nich. Konkrétny variant považujeme za dostatočne podporený ak platí $m \geq a, m \geq nr$. Takýto variant potom vypíšeme na výstup, kým varianty ktoré nesľňajú túto podmienku vyfiltrujeme.

Parameter p je počet pozícií, o ktoré budeme každý interval rozširovať doprava a doľava.

Navrhnutý algoritmus FB budeme v tejto kapitole porovnávať s priamočiarym algo-

ritmom z prvej kapitoly, ktorý ďalej budeme nazývať pileup. Tento algoritmus má iba parametre a, r a majú význam ako pri FB algoritme.

2.2 Dáta pre účely vyhodnotenia

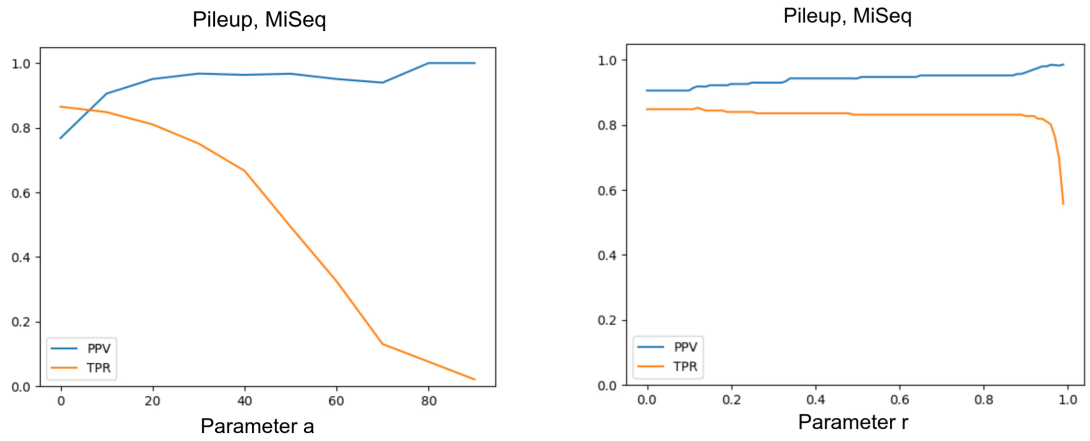
Pre účely vyhodnotenia popísaných prístupov sme používali dve vzorky dát.
Prvá vzorka pochádza z baktérie *Eschericia Coli* a dáta boli sekvenované pomocou technológie MiSeq [13]. Použili sme vzorku 10000 čítaní, ktoré pokrývajú oblasť genómu dĺžky 21038 pozícií. Priemerná dĺžka čítania je 131, priemerné pokrytie každej pozície v referenčnej sekvencii je 61. Genóm z tejto vzorky sme získali z NBCI (angl. National Center for Biotechnology Information) pod menom `NZ_CP026026.1`. Čítania z tejto vzorky sme získali z prehliadača ENA (angl. European nucleotide archive) pod id `ERX008638`. Varianty ktoré sme potom považovali za správne sme získali pomocou nástroja Bcftools. Čítania získané pomocou technológie MiSeq sú veľmi presné, s chybovosťou menšou než 1 %, preto je problém idetifikácie variantov pomocou tejto technológie ľahší.

Druhá vzorka pochádza zo sekvencie vírusu SARS-COV-2 sekvenovanej pomocou technológie MinION, v Biomedicínskom centre Slovenskej akadémie vied. Genóm vírusu má dĺžku 29903 báz, použili sme 70637 čítaní, pričom priemerná dĺžka čítania je 516 báz. Priemerné pokrytie je 925. Vzorovú odpoveď sme získali za pomoci štandardých metód analýzy prostredníctvom softvéru ArticPipeline [15]. Dáta získané nanopórovým sekvenovaním sú náročnejšie na analýzu, keďže čítania majú pomerne veľkú chybovosť a obsahujú veľké množstvo inzercií a delécií, ktoré sťažujú zarovnanie sekvencií ku referncii. Vlastnostiam nanopórových dát sa budeme bližšie venovať v kapitole 3.

2.3 Porovnanie úspešnosti algoritmov pileup a FB

Výsledky oboch algoritmov budeme vyhodnocovať podľa dvoch indikátorov:
TPR (z anglického true positive rate) je podiel počtu všetkých správnych nájdených variantov a počtu všetkých variantov ktoré sme mali nájsť. Toto číslo vypovedá o tom, akú časť správnych variantov sa nám podarilo objaviť.
PPV (z anglického positive predictive value) je podiel počtu všetkých správnych nájdených variantov voči všetkým nájdeným variantom. Toto číslov vypovedá o tom, akú časť všetkých nájdených variantov tvorili správne varianty.

Na dátach zo sekvenačnej platformy MiSeq dosiahli obidva algoritmy približne rovnaké výsledky, viď obrázok 2.1,2.2. Obidvom algoritmom sa podarilo nájsť 80 percent



Obr. 2.1

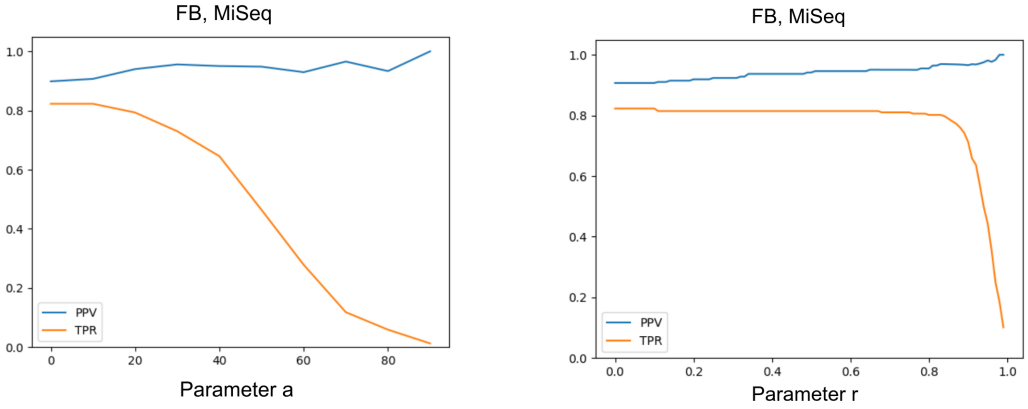
hľadaných variantov s vysokou presnosťou. Podobnosť výsledkov oboch algoritmov je spôsobená tým, že čítania z prvej vzorky obsahovali predovšetkým bodové rozdiely-oproti referenčnej sekvencii. Pri takýchto dátach je možné veľmi spoľahlivo zarovnať sekvencie ku referencii a oba algoritmy fungujú v tomto kontexte veľmi podobným spôsobom.

Na obrázkoch 2.3, 2.4 sú zobrazené výsledky oboch algoritmov na vzorke zo sekvenačnej platformy MinION. Na výsledkoch z tejto vzorky sa dali pozorovať isté rozdiely medzi jednotlivými prístupmi.

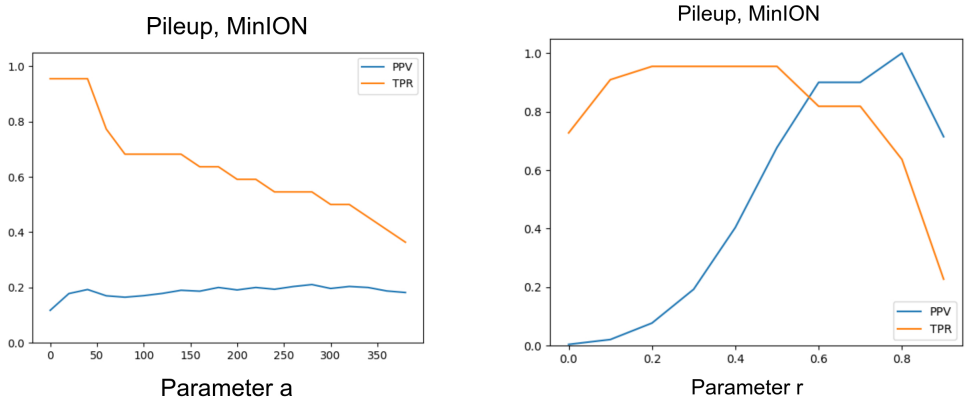
Pileup prístup dosahoval pri nízkych hodnotách vstupných parametrov vysoké hodnoty TPR a nízke hodnoty PPV. To znamená že, algoritmus nájde takmer všetky správne varianty, ale aj veľa nesprávnych. Zvýšením hodnôt vstupných parametrov dosiahneme zlepšenie hodnoty PPV, ale zároveň sa zníži hodnota TPR. Nízke hodnoty PPV sú spôsobené veľkým množstvom nájdených delécií, ktoré neboli správne. Príčinou tohoto javu je veľká nepresnosť čítaní získaných pomocou metódy nanopórového sekvenovania.

Za pomoci FB prístupu sa podarilo tomuto problému vyhnúť a dosiahnuť zároveň vysoké hodnoty PPV aj TPR.

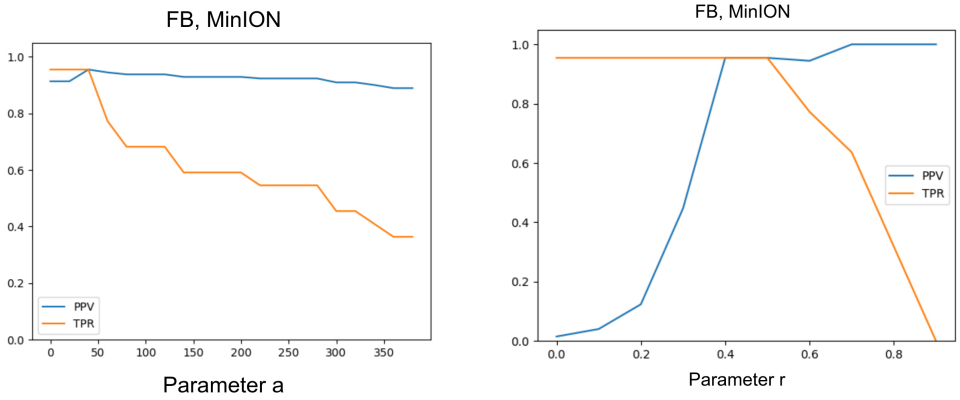
Pre porovnanie sme vyhodnotili výsledky oboch algoritmov aj v prípade, že sme neuvažovali delécie. Na obrázku 2.5 vidíme, že hodnota PPV vo výsledku pileup algoritmu je výrazne lepšia, zatiaľ čo hodnota TPR sa prakticky nezmenila. Naproti tomu obrázok 2.6 ukazuje, že výsledky FB algoritmu sa nezmenili, teda FB je vhodný aj na detekciu delécií. Zároveň, v druhej vzorke sa nenachádzali žiadne inzercie, preto sme na troch vopred určených miestach zmazali z referenčnej postupnosti niekoľko znakov a čítania zarovnali k takto vzniknutej novej referenčnej postupnosti. Takto sme vlastne umelo vytvorili vo vzorke inzercie. Pileupu sa podarilo nájsť iba dve z týchto umelo vytvorených inzercií, zatiaľ čo FB našiel všetky tri.



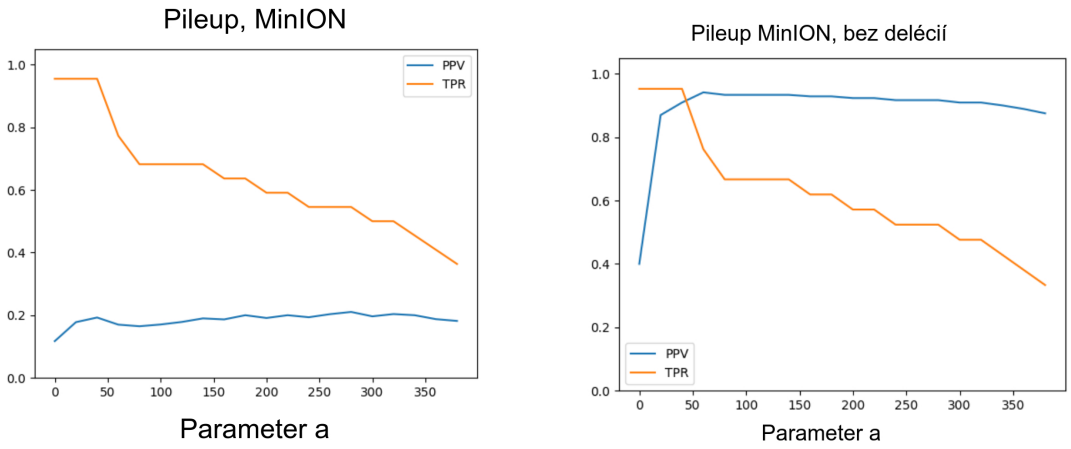
Obr. 2.2



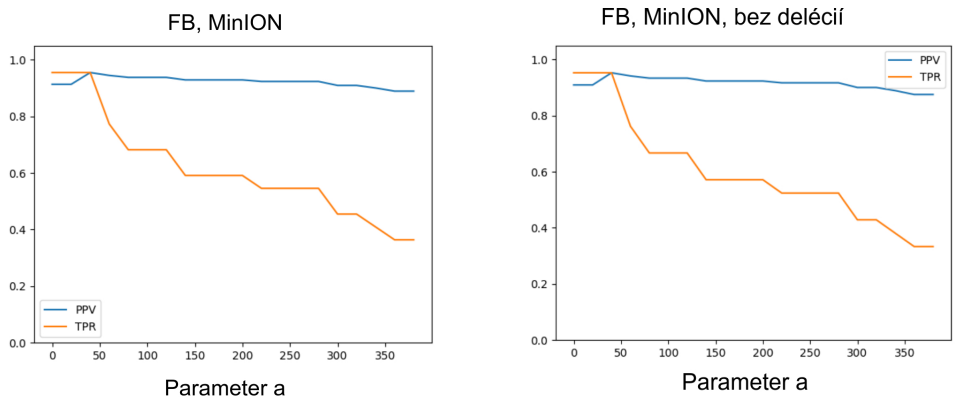
Obr. 2.3



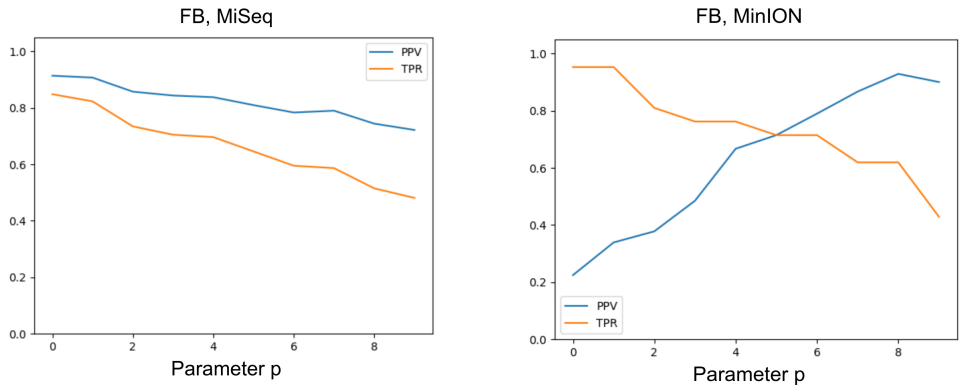
Obr. 2.4


Pileup, MinION
PPV
TPR
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250
300
350
Parameter a
Pileup MinION, bez delécií
PPV
TPR
Parameter a

Obr. 2.5


FB, MinION
PPV
TPR
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250
300
350
Parameter a
FB, MinION, bez delécií
PPV
TPR
Parameter a

Obr. 2.6



Obr. 2.7

r	a	PPV	TPR
0.5	30	0.645	0.909
0.5	40	0.645	0.909
0.55	10	0.714	0.909
0.55	20	0.741	0.909
0.55	30	0.769	0.909
0.55	40	0.769	0.909

Tabuľka 2.1: V tejto tabuľke sa nachádzajú vyfiltrované najlepšie výsledky pileup algoritmu z druhej vzorky.

r	a	p	PPV	TPR
0.400	30	1	0.955	0.955
0.400	40	1	0.955	0.955
0.450	30	1	0.955	0.955
0.450	40	1	0.955	0.955

Tabuľka 2.2: V tejto tabuľke sa nachádzajú vyfiltrované najlepšie výsledky FB algoritmu z druhej vzorky.

V tabuľkách 2.2, 2.1 sú na porovnanie uvedené vyfiltrované najlepšie výsledky oboch algoritmov z druhej vzorky. Jasne vidno, že pomocou FB sa dajú na týchto dátach dosiahnuť lepšie hodnoty TPR a PPV ako pomocou pileupu.

Na obrázku 2.7 je zobrazená závilosť hodôt TPR, PPV od parametra p pri obidvoch vzorkách. Klesavý trend hodnoty TPR v závislosti od hodnoty p, ktorý je silnejší pri druhej vzorke, je spôsobený tým, že keď zvýšime hodnotu p a teda aj dĺžku okna, je väčšia pravdepodobnosť, že dostaneme v tomto okne pri rôznych čítaniach rôzne reťazce. Tým sa odfiltrujú niektoré varianty, ktoré boli správne a zníži sa hodnota TPR.

Záverom, na čítaniach s vysokou spoľahlivosťou bez inzercií a delécií majú obidva testované algoritmy približne rovnakú úspešnosť. Na dátach získaných pomocou nanopórového sekvenovania, ktoré majú nižšiu spoľahlivosť, je pileup algoritmus vhodný primárne na detekciu SNPov. FB je aj pri takýchto dátach vhodný na hľadanie SNPov, delécií a inzercií.

Kapitola 3

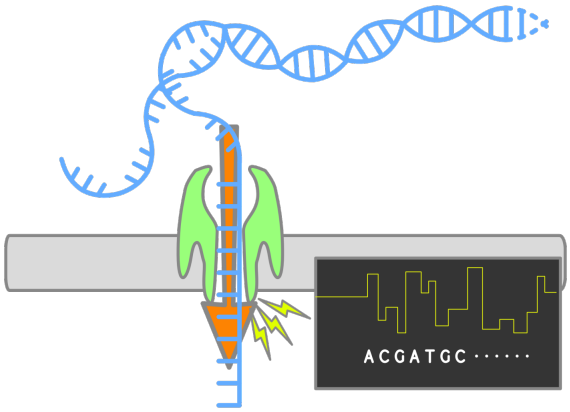
Špecifické vlastnosti nanopórového sekvenovania

Nanopórové sekvenovanie je nová metóda sekvenovania, ktorá má mnoho výhod: dlhé čítania, rýchlosť a nízka cena. Nevýhodou je, že v takto získaných dátach sa nachádza veľa nepresností.
Princíp nanopórového sekvenovania je nasledovný: v membráne, ktorá sa nachádza medzi dvoma komorami s elektrolytom, sa nachádza malý otvor, nanopór. V jednej z komôr sa nachádza vzorka DNA. Rozdiel elektrických potenciálov medzi jednotlivými komorami vytvorí elektrické napätie. Vzniknutá sila ťahá vlákno DNA cez nanopór. Bázy, ktoré sa práve nachádzajú v blízkosti nanopóru ovplyvňujú hodnotu elektrického prúdu, ktorá tečie cez nanopór, pričom každá z báz ovplyvňuje hodnotu elektrického prúdu špecifickým spôsobom, viď obrázok 3.1. Zo zmeny elektrického prúdu sme schopní detekovať aká báza sa nachádzala v nanopóre. Problém je, že prechod vlákna DNA cez nanopór nie je nutne lineárny. Zmeny v rýchlosti a smere pohybu vlákna výrazne sťažujú spracovanie signálu a spôsobujú spomínané nepresnosti.

Príkladom prístroja, ktorý umožňuje realizovať nanopórové sekvenovanie je MinION sekvenátor [12] (obrázok 3.2) vyrábaný britskou firmou Oxford Nanopore Technologies. Výstupom MinION prístroja sú hodnoty elektrického prúdu, ktoré budeme ďalej nazývať surový signál (obrázok 3.3). Medzi jeho výhody patria dostupnosť a jednoduchosť realizácie sekvenovania.

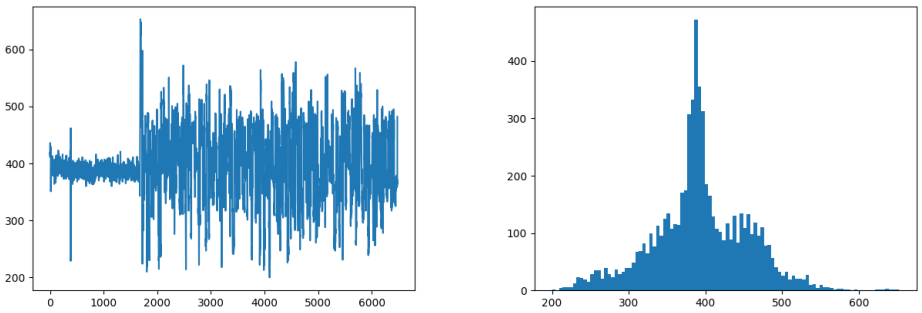
3.1 Určovnie báz pomocou neurónových sietí a vrstva CTC

Preložiť postupnosť hodnôt elektrického prúdu do postupnosti báz DNA (reťazca nad abecedou $\{A, C, G, T, -\}$) je netriviálna úloha, ktorá sa rieši pomocou hlbokých neuró-



Obr. 3.1: Prechod DNA vlákna cez nanopór.
Zdroj: `https://miro.medium.com/max/3840/0*glhxCHhF3SBOsMYa.png`

Obr. 3.2: Minion sekvenátor
Zdroj: `https://qtxasset.com/fiercebiotech/1539178811/photo%2051880840.jpg?HBKx2jrpwrshc7Ty.ST76GyG996IN_SG`



Obr. 3.3: Graf a rozdelenie hodnôt signálu z jedného čítania.

nových sietí - viď napríklad DeepNano-blitz.

Signál najprv upravíme pomocou mediánovej normalizácie. Prvá časť DeepNano-blitz obsahuje niekoľko neurónových vrstiev rôznych typov. Pre našu prácu je podstatná CTC vrstva [3], ktorá je jedna z posledných.

CTC je metóda používaná v prípade, keď potrebujeme istej postupnosti priradiť kratšiu postupnosť znakov z konečnej množiny a zarovnanie vstupnej a výstupnej postupnosti je neznáme. Metóda bola uvedená Gravesom [10] a medzi príklady použitia patrí rozoznávania reči [22] a písma [11].

Nech X je vstupná postupnosť, S je konečná množina znakov, znak $-$ je takzvaný blank. Vstupom do CTC vrstvy neurónovej siete je softmax vrstva, ktorá pre každý člen postupnosti X vyprodukuje $|S| + 1$ pravdepodobností. Tieto pravdepodobnosti predstavujú pravdepodobnostné rozdelenie nad množinou $S \cup \{-\}$.

Nech na vstupe neurónovej siete je vstupný signál ako na obrázku 3.6, potom vstup pre CTC vrstvu bude pre každú pozíciu vstupného signálu obsahovať 5 hodnôt reprezentujúcich pravdepodobnosti písmen A, C, G, T a $-$. Informáciu môžeme vizualizovať ako na obr. 3.7. Pravdepodobnosť písmena i na pozícii j označme $p_{i,j}$.

Každej možnej postupnosti znakov $\{A, C, G, T, -\}^n$, kde n je dĺžka vstupnej postupnosti, vieme na základe hodnôt v tejto vrstve priradiť priradiť pravdepodobnosť. Konkrétne postupnosti $x = x_1...x_n$ priradíme pravdepodobnosť $P_{x_{i=1}^n} = \prod p_{i,x_i}$.

Výsledkom CTC vrstvy je kratšia postupnosť znakov z postupnosti X môžeme vyrobiť redukovanú postupnosť $red(x)$, tak že najprv všetky za sebou idúce rovnaké znaky zredukujeme do jedného písmena, a potom z výslednej postupnosti vylúčime znaky $-$. Napríklad $red(AAACCG - GT) = ACGGT$. CTC vrstva zo všetkých možných redukovaných reťazcov vyberie taký reťazec Y, ktorý má najväčšiu možnú pravdepodobnosť q_Y.

Všimnime si, že viacero reťazcov môže mať rovnaký redukovaný reťazec. Jednododuchý príklad: red(ATT-TCG) = red(AAT-TTCGG) = $ATTCG$.

Potom

$$q_Y = \sum_{X:red(X)=Y} p_X$$

Takýmto spôsobom dokáže CTC vrstva vyrobiť z dlhšieho signálu kratší výsledný reťazec, pričom počet vstupných hodnôt zodpovedajúcich jednej výstupnej hodnote je premenlivý.

Výhodou tejto metódy je, že na začiatku nepotrebuje poznať vzájomné zarovnanie vstupnej a výstupnej postupnosti.

3.2 Vzorkovanie z CTC vrstvy

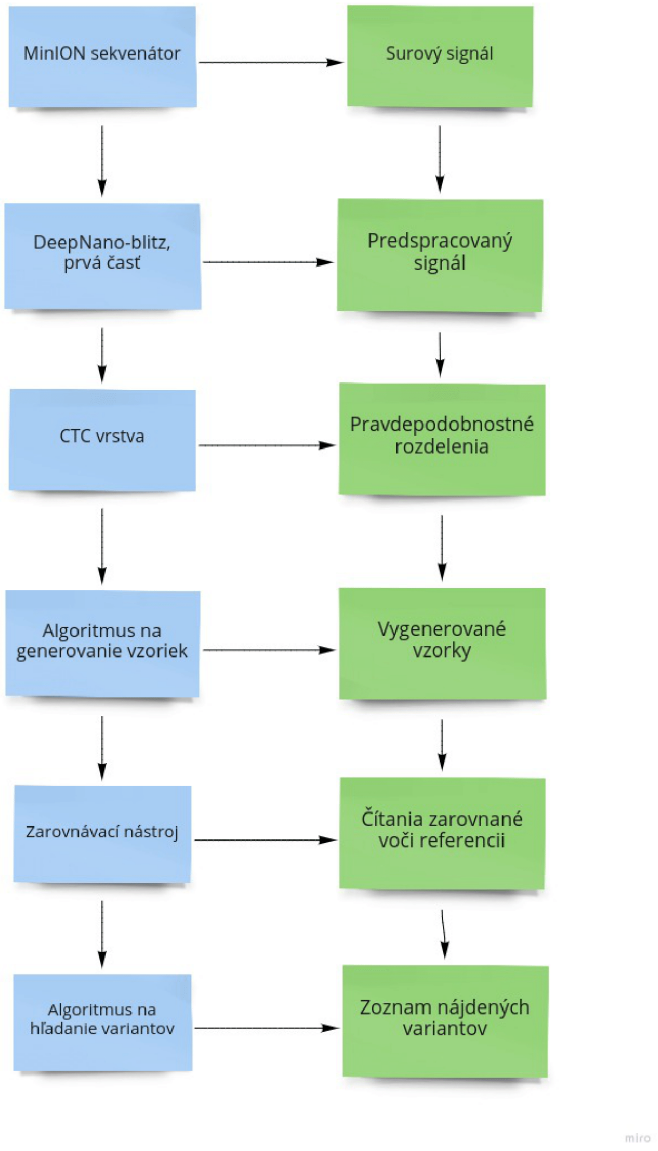
Do zdrojového kódu DeepNano-blitz sme doplnili funkcionalitu, ktorá nám umožnila extrahovať vstup CTC vrstvy, teda tabuľku p_{ij}, ktorá predstavuje pre každú vstupnú pozíciu i pravepodobnostné rozdelenie nad množninou $\{A, C, G, T, -\}$.
Uvažujme nasledovný postup:

1. Predpokladajme, že sme už získali dáta zo vzorky, pomocou sekvenátora MinION.
2. Pomocou DeepNano-blitz predspracujeme signál pre CTC vrstvu.
3. Pomocu CTC vrstvy vygenerujeme opísané pravdepodobnostné rozdelenie p_{ij}.
4. Z týchto rozdelení náhodne vygenerujeme reťazec zo znakov $A, C, G, T, -$.
5. Tento reťazec zredukujeme - odstránime všetky za sebou nasledujúce rovnaké znaky, potom odstránime všetky výskyty znaku $-$.
6. Výsledok budeme nazývať vzorkou.

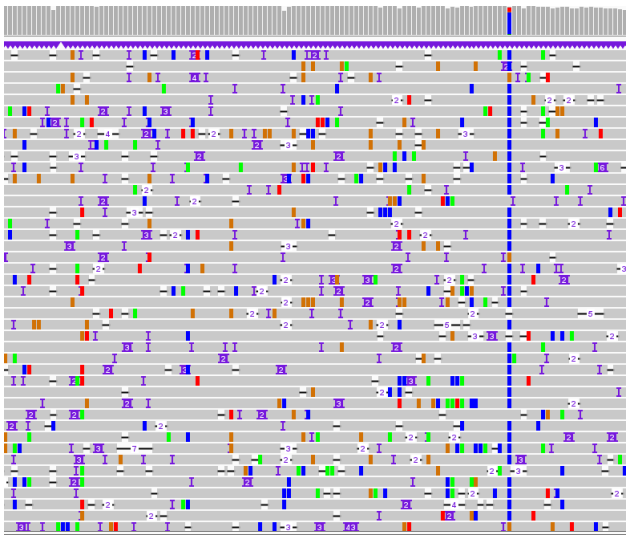
Týmto spôsobom získame viacero vzoriek, ktoré predstavujú niekoľko možných určení báz pre daný signál získaný zo sekvenátora MinION. Príklad výsledku vzorkovania v kroku 4. vidíme na obrázku 3.8, po zredukovaní na obrázku číslo 3.9.
Výhodou tohto prístupu je, že z jedného čítania si vieme vygenerovať ľubovoľne veľa vzoriek. So vzorkami budeme ďalej narábať rovnako ako s čítaniami.

Na obrázku 3.4 sú zobrazené jednotlivé procesy ako za sebou nasledujú počas vzorkovania a následného hľadania variantov. Ku každému procesu je priradený jeho výstup, ktorý slúži ako vstup do nasledujúceho procesu.

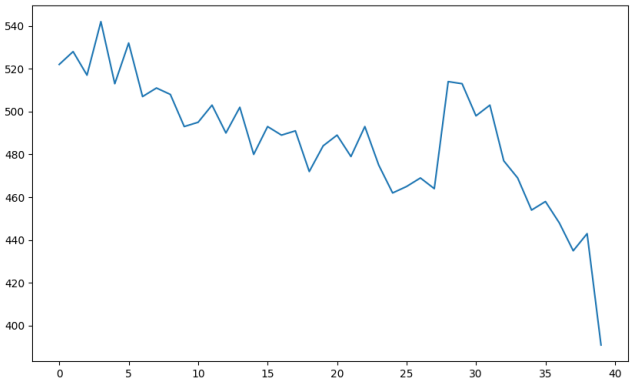
Na obr. 3.5 je 100 vzoriek vygenerovaných pomocou postupu opísaného vyššie, zarovananých voči referencii a zobrazených pomocou nástroja IGV [19]. Výrazný modrý stĺpec na pozícii 14458 reprezentuje jednonukleotidový polymorfizmus, ktorý sa na tom mieste nachádza.



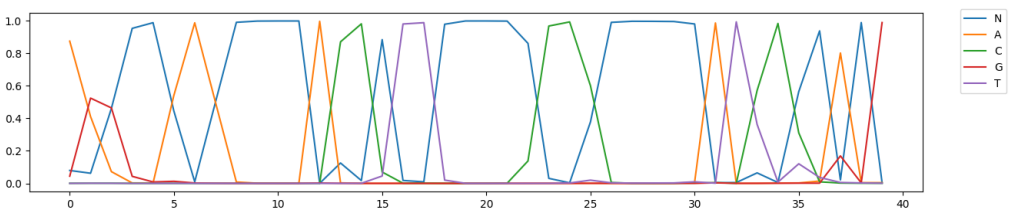
Obr. 3.4: Za sebou nasledujúce procesy v algoritme vzorkovania a následného hľadania variantov



Obr. 3.5: Vygenerované vzorky z reálneho nanopórového čítania.



Obr. 3.6: Úsek surového signálu z jedného čítania dlhý 20 pozícií



Obr. 3.7: Pravdepodobnostné rozdelenia k signálu 3.6

AG- - -AAA- - - -A-C-TTT- - - -CCC- - - - -ATCC- -A-G
GG- - - -AA- - - -ACC-TT- - - - -CCC- - - - -ATTC- -A-G
AG- - - -AA- - - -ACC-TT- - - - -CC- - - - - -ATCCTTA-G
AA- - - -AA- - - -ACC-TT- - - - -CCC- - - - -ATCC- -A-G
AAG- - -AA- - - -ACC-TT- - - - -CCC- - - - -ATCCC-A-G
-A- -A-AA- - - -ACC-TT- - - - -CCC- - - - -ATCCC-A-G
-AG- - -A- - - - -ACCCTT- - - - -CCC- - - - -ATTC- -A-G
AGG- - -AA- - - -ACCTTT- - - - -CCC- - - - -ATTC- -A-G
A- - - -AA- - - - -ACC-TT- - - - -CCC- - - - -ATCC- -A-G
AA- - -AAA- - - -ACC-TT- - - - -CC- - - - - -ATTCG-A-G

Obr. 3.8: 10 reťazcov vygenerovaných z rozdelení na obrázku 3.7

AGAACTCCATCTG
AGAACTCATCAG
GAACTCATCAG
AGAACTCATCGG
AAACTCATCAG
AAACTCATCAG
GAAACTCATCAG
AAACTCATCAG
AAACTCTATCAG
AAACTCATCAG

Obr. 3.9: Výsledné vzorky po zredukovaní reťazcov z obrázka 3.8

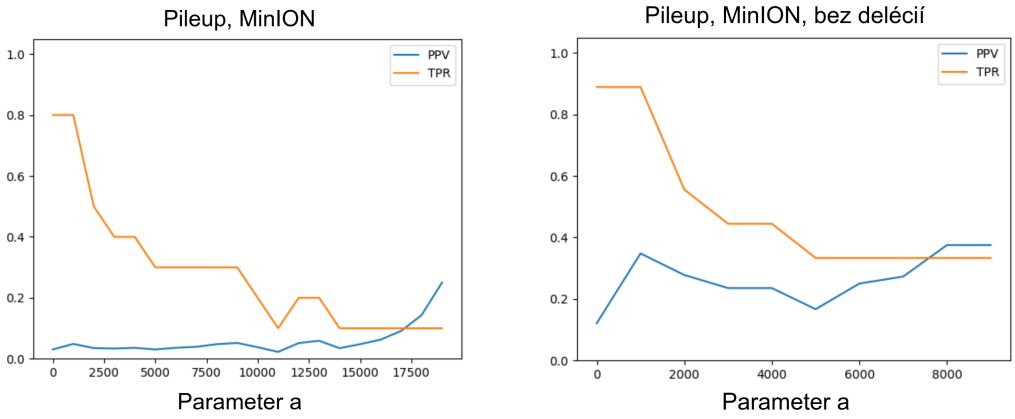
Kapitola 4

Vplyv vzorkovania pri hľadaní variantov

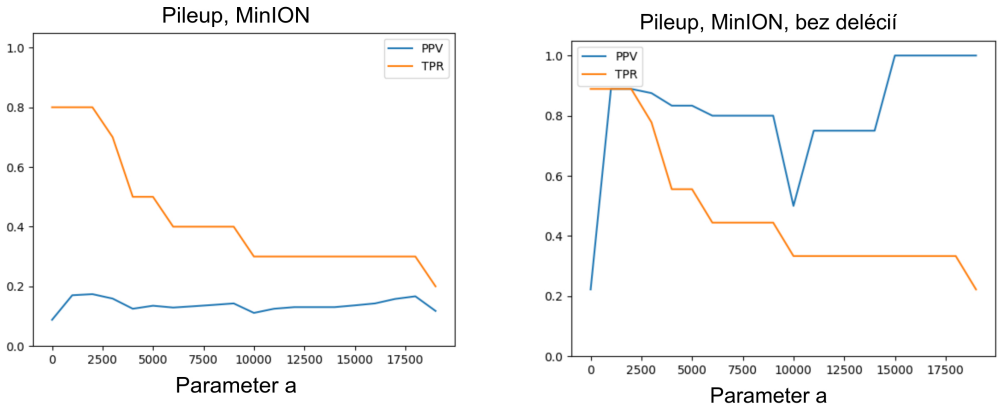
V tejto kapitole budeme skúmať vplyv vzorkovania na výsledky algoritmov popísaných v predchádzajúcich kapitolách. Naša hypotéza je, že vzorkovanie dokáže zlepšiť výsledky týchto algoritmov.
Predpokladajme, že na vstupe dostaneme zoznam čítaní a referenciu. Uvažujme nasledovný postup: Ku každému čítaniu vygenerujeme n vzoriek pomocou DeepNanoblitz. Vygenerované vzorky zarovnáme vzhľadom k referencii a aplikujeme na ne náš algoritmus. Na overenie našej hypotézy sme použili skupinu 15000 čítaní zo vzorky SARS-COV-2, ktorú sme opísali v kapitole 2.

4.1 Vplyv vzorkovania na výsledky pileup prístupu

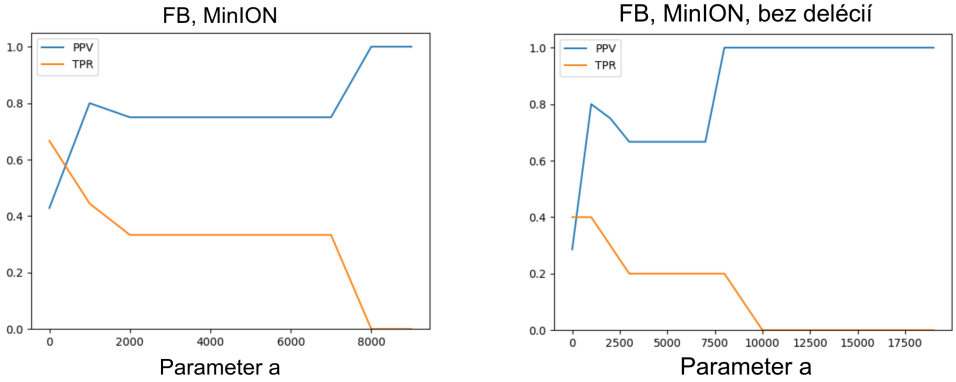
Ako možno vidieť z obrázkov 4.1 a 4.2, pomocou vzorkovania sa pileup algoritmu podarilo dosiahnuť takmer rovnako dobré výsledky ako s použitím obyčajných čítaní.



Obr. 4.1



Obr. 4.2

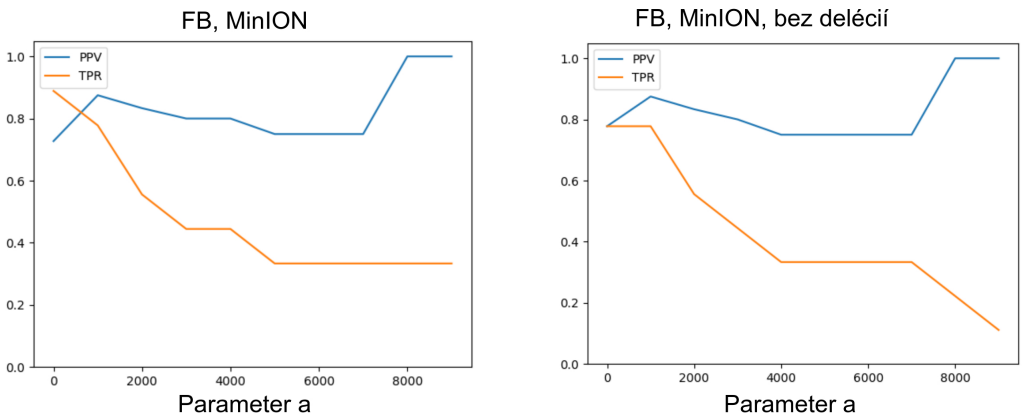


Obr. 4.3

Príčinou nízkej hodnoty PPV bolo opäť veľké množstvo nesprávnych delécií. Preto sme opäť vyhodnotili aj hypotézu s ignorovaním delécií. Z obrázkov tiež vidno, že ignorovanie delécií môže značne zvýšiť hodnotu PPV.

4.2 Vplyv vzorkovania na výsledky FB prístupu

Z obrázku 4.4 vidno, že pomocou vzorkovania sa FB prístupu podarilo dosiahnuť takmer rovnako dobré hodnoty TPR a PPV ako pomocou priamočiareho prístupu. Zároveň z obrázku 4.3 možno usúdiť, že pokiaľ nie je hodnota n dostatočne vysoká, FB nedosahuje tak dobré hodnoty TPR ako priamočiary prístup. Ignorovanie delécií nezmenilo hodnoty TPR a PPV v prípade FB algoritmu.


FB, MinION
PPV
TPR
1.0
0.8
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
Parameter a
FB, MinION, bez delécií
PPV
TPR
1.0
0.8
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
Parameter a

Obr. 4.4

Kapitola 5

Implementačné detaily

V tejto kapitole podrobnejšie vysvetlíme, ako fungoval proces vzorkovania a následného hľadania variantov. V našej práci sme používali veľa rôznych nástrojov, knižníc a súborových formátov. V tejto kapitole tiež podrobne vysvetlíme načo nám slúžili.

5.1 Použité programovacie jazyky

Obidva opísané algoritmy sme implementovali v jazyku Python. Je to interpretovaný, dynamicky typovaný jazyk. Medzi jeho výhody patrí rýchlo rastúca učiaca krivka, čitateľnosť kódu a bohatá komunita, vďaka ktorej existuje v Pythone množstvo knižníc. Jeho hlavnou nevýhodou je, že je relatívne pomalý oproti iným jazykom ako je C++ alebo Java. Toto nám ale v našej práci neprekážalo, pretože sme sa nesústredili na výkon.

Funkcionalitu, ktorá umožnila extahovať výstup CTC vrstvy sme naprogramovali v jazyku Rust. Rust je pomerne nový, kompilovaný a staticky typovaný jazyk, ktorý sa sústredi na bezpečnosť a odstraňovanie možných chýb už počas kompilácie. Vyznačuje sa vysokou rýchlosťou a spoľahlivosťou. Nevýhodou je, že je omnoho náročnejší na učenie.

5.2 Použité súborové formáty a softvérové nástroje

V bioinformatike sa používa veľké množstvo rôznych typov dát, ktoré si vyžadujú rôzne súborové formáty.

Surový signál z MinION sekvenátora sa ukladá v súborovom formáte `fast5`. Na pracovanie s takýmito dátami sme používali knižnicu `ont_fast5_api`.
Na ukladanie DNA postupností je možné použiť niekoľko rôznych formátov. My sme používali formát `fasta` pre jeho prehľadnosť a jednoduchosť. Na prácu s týmto formá-

tom sme používali knižnicu `fastaparser` v jazyku Python.
Na ukladanie zarovnaných DNA postupností sa používajú formáty `SAM` a `BAM`. Na prácu s nimi sme používali knižnicu `pysam`. Na zarovnávanie čítaní voči referencii sme používali nástroje samtools a minimap2.
Na ukladanie variantov slúži súborový formát `VCF`. Parsovanie tohto formátu je relatívne jednoduché, preto sme to naprogramovali sami.
Na získanie variantov, ktoré sme potom považovali za správne odpovede sme použili nástroj bcftools.

Záver

V tejto práci sme implementovali dva algoritmy na hľadanie variantov, ktoré sme následne otestovali na dátach zo sekvenačnej platformy MinION a MiSeq a porovnávali sme ich výhody a nevýhody. Zistili sme, že na dátach s vysokou spoľahlivosťou (MiSeq) majú obidva algoritmy približne rovnakú úspešnosť. Na dátach, ktoré obsahujú veľa nepresností (MinION) má FB algoritmus výrazne vyššiu úspešnosť, pretože berie do úvahy aj zarovnanú postupnosť. Vďaka tomu je vhodný aj na detekciu inzercií a delécií. Pileup algoritmus našiel priveľké množstvo delécií, ktoré neexistovali, teda nie je vhodný na takýto typ dát.
Tiež sme skúmali hypotézu, že dodatočná informácia z CTC vrstvy by mohla zlepšiť úspešnosť týchto algoritmov. Túto hypotézu sa nám podarilo potvrdiť iba čiastočne, keďže sme dosiahli takmer rovnaké výsledky ako pri priamočiarom prístupe. Pileup opäť našiel veľké množstvo neexistujúcich delécií. Na druhej strane tento prístup fungoval dostatočne dobre pri detekcii SNPov. Pomocou FB prístupu sa podarilo dosiahnuť vysoké hodnoty TPR a PPV. Treba ale povedať že tieto výsledky sa podarilo dosiahnuť, len ak sme vygenerovali dostatočne veľa vzoriek z jedného čítania.

Možností ako pokračovať v tejto práci je niekoľko.
Z praktických dôvodov by bolo dobré implementovať obidva spomínané algoritmy v rýchlejšom jazyku (pravdepodobne C++ alebo Rust). CTC vrstva má nevýhodu, pretože predpokladá, že jednotlivé pozície sú navzájom nezávislé. Namiesto CTC vrstvy by sme mohli použiť takzvanú RNA neurónovú sieť (angl. Recurrent Neural Aligner) [18]. RNA neurónová sieť pracuje s predpokladom, že jednotlivé pozície od seba môžu závisieť, a teda týmto spôsobom by sa možno dali dosiahnuť ešte lepšie výsledky pri vzorkovaní.

Literatúra

[1] Irantzu Anzar, Angelina Sverchkova, Richard Stratford, and Trevor Clancy. Neomutate: an ensemble machine learning framework for the prediction of somatic mutations in cancer. *BMC Medical Genomics*, 12(1):63, May 2019.

[2] Mohammad Reza Bakhtiarizadeh and Ali A. Alamouti. Rna-seq based genetic variant discovery provides new insights into controlling fat deposition in the tail of sheep. *Scientific Reports*, 10(1):13525, Aug 2020.

[3] Vladimír Boža, Peter Perešíni, Broňa Brejová, and Tomáš Vinař. Deepnano-blitz: A fast base caller for minion nanopore sequencers. *bioRxiv*, 2020.

[4] Phelim Bradley, N. Claire Gordon, Timothy M. Walker, Laura Dunn, Simon Heys, Bill Huang, Sarah Earle, Louise J. Pankhurst, Luke Anson, Mariateresa de Cesare, Paolo Piazza, Antonina A. Votintseva, Tanya Golubchik, Daniel J. Wilson, David H. Wyllie, Roland Diel, Stefan Niemann, Silke Feuerriegel, Thomas A. Kohl, Nazir Ismail, Shaheed V. Omar, E. Grace Smith, David Buck, Gil McVean, A. Sarah Walker, Tim E.A. Peto, Derrick W. Crook, and Zamin Iqbal. Rapid antibiotic resistance predictions from genome sequence data for s. aureus and m. tuberculosis. *bioRxiv*, 2015.

[5] Lei Cai, Yufeng Wu, and Jingyang Gao. Deepsv: accurate calling of genomic deletions from high-throughput sequencing data using deep convolutional neural network. *BMC Bioinformatics*, 20(1):665, Dec 2019.

[6] Petr Danecek and Shane A McCarthy. BCFtools/csq: haplotype-aware variant consequences. *Bioinformatics*, 33(13):2037–2039, 02 2017.

[7] Robert C. Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, Mar 2004. 15034147[pmid].

[8] Erik Garrison and Gabor Marth. Haplotype-based variant detection from short-read sequencing. *arXiv*, 1207, 07 2012.

[9] Manolo Gouy, Stéphane Guindon, and Olivier Gascuel. SeaView Version 4: A Multiplatform Graphical User Interface for Sequence Alignment and Phylogenetic Tree Building. *Molecular Biology and Evolution*, 27(2):221–224, 10 2009.

[10] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA, 2006. Association for Computing Machinery.

[11] Alex Graves, Santiago Fernández, Marcus Liwicki, Horst Bunke, and Jürgen Schmidhuber. Unconstrained on-line handwriting recognition with recurrent neural networks. volume 20, 01 2007.

[12] Miten Jain, Hugh E. Olsen, Benedict Paten, and Mark Akeson. The oxford nanopore minion: delivery of nanopore sequencing to the genomics community. *Genome Biology*, 17(1):239, Nov 2016.

[13] George John Kastanis, Luis V. Santana-Quintero, Maria Sanchez-Leon, Sara Lomonaco, Eric W. Brown, and Marc W. Allard. In-depth comparative analysis of illumina(®) miseq run metrics: Development of a wet-lab quality assessment tool. *Molecular ecology resources*, 19(2):377–387, Mar 2019. 30506954[pmid].

[14] Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 05 2018.

[15] Jun Li, Haoqiu Wang, Lingfeng Mao, Hua Yu, Xinfen Yu, Zhou Sun, Xin Qian, Shi Cheng, Shuchang Chen, Junfang Chen, Jingcao Pan, Jueliang Shi, and Xuchu Wang. Rapid genomic characterization of sars-cov-2 viruses from clinical specimens using nanopore sequencing. *Scientific Reports*, 10(1):17492, Oct 2020.

[16] Aaron McKenna, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly, and Mark A. DePristo. The genome analysis toolkit: a mapreduce framework for analyzing next-generation dna sequencing data. *Genome research*, 20(9):1297–1303, Sep 2010. 20644199[pmid].

[17] Ryan Poplin, Pi-Chuan Chang, David Alexander, Scott Schwartz, Thomas Colthurst, Alexander Ku, Dan Newburger, Jojo Dijamco, Nam Nguyen, Pegah T. Afshar, Sam S. Gross, Lizzie Dorfman, Cory Y. McLean, and Mark A. DePristo. A universal snp and small-indel variant caller using deep neural networks. *Nature Biotechnology*, 36(10):983–987, Nov 2018.

[18] H. Sak, M. Shannon, K. Rao, and F. Beaufays. Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping. In *INTERSPEECH*, 2017.

[19] Helga Thorvaldsdóttir, James T. Robinson, and Jill P. Mesirov. Integrative genomics viewer (igv): high-performance genomics data visualization and exploration. *Briefings in bioinformatics*, 14(2):178–192, Mar 2013. 22517427[pmid].

[20] Andrea D. Tyler, Laura Mataseje, Chantel J. Urfano, Lisa Schmidt, Kym S. Antonation, Michael R. Mulvey, and Cindi R. Corbett. Evaluation of oxford nanopore's minion sequencing device for microbial whole genome sequencing applications. *Scientific Reports*, 8(1):10931, Jul 2018.

[21] Jianmin Wang and Xiaoqiu Huang. A method for finding single-nucleotide polymorphisms with allele frequencies in sequences of deep coverage. *BMC Bioinformatics*, 6(1):220, Sep 2005.

[22] Takenori Yoshimura, Tomoki Hayashi, Kazuya Takeda, and Shinji Watanabe. End-to-end automatic speech recognition integrated with ctc-based voice activity detection, 2020.