VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2023 David Pešek

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

UMĚLÁ NEURONOVÁ SÍŤ PRO REKONSTRUOVÁNÍ VYMŘELÝCH DRUHŮ

A NEURAL NETWORK FOR RECONSTRUCTION OF EXTINCT ANIMALS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — David Pešek

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Václav Jirsík, CSc.

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: David Pešek ***ID:*** 228732

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Umělá neuronová síť pro rekonstruování vymřelých druhů

POKYNY PRO VYPRACOVÁNÍ:

1. Proveďte rešerši modelů umělých neuronových sítí a vyberte nejvhodnější pro zero-shot rekonstrukci druhů.
2. Vytvořte trénovací a testovací množinu rozdělené podle taxonomického druhu.
3. Naučte vybranou neuronovou síť na vytvořené trénovací množině.
4. Porovnejte vámi vygenerované zero-shot rekonstrukce s testovací množinou.

DOPORUČENÁ LITERATURA:

Hierarchical Text-Conditional Image Generation with CLIP Latents, https://doi.org/10.48550/arXiv.2204.06125

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 22.5.2023

Vedoucí práce: doc. Ing. Václav Jirsík, CSc.

Konzultant: Ing. Dominik Řičánek

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

ABSTRAKT

Tato práce se zabývala navrhnutím, naučením a zhodnocením umělé neuronové sítě pro rekonstrukci vymřelých živočišných druhů. Nejprve byl vybrán hlavní prvek navrhované UNS, tedy generativní model. Vzhledem k jejich výborným výsledkům v poli generování obrázků se odůvodněně jevila třída difúzních modelů jako správná volba. Konkrétně byl vybrán difúzí model *Stable diffusion*[1].

Jeden z počátečních kroků práce bylo také vytvořit trénovací množinu pro navrhovaný model. K obrázkům živočichů bylo potřeba napárovat nějaké popisky, podle kterých by se dal živočich identifikovat. K tomuto účelu byly využity geny cyklooxygenázy-1 daných živočichů.

Dále byl použit sekvenční transformátorový model GPT-2[2], který je naučen na trénovací množině lidského přirozeného jazyka. Tento model byl použitý pro zakódování DNA sekvencí do vektorové podoby, ve které byla zachycena sémantika a kontext mezi jednotlivými částmi DNA sekvence.

Modely by bylo velmi složité učit od začátku kvůli velké potřebné velikosti trénovací množiny a výpočetní a časové náročnosti. GPT-2 model byl tedy pouze doučen na trénovací množině DNA sekvencí řádu pěvců a samotný difúzní model byl naučen na párech obrázků těchto živočichů a DNA sekvencí zakódovaných pomocí GPT-2 modelu.

Pro generování obrázků byly pomocí GPT-2 generovány originální DNA sekvence, které se podobaly sekvencím z trénovací množiny. Následně bylo zakódování těchto sekvencí předáno difúznímu modelu, který vytvořil samotné obrázky. Metoda generování nových DNA sekvencí pomocí GPT-2 modelu stojí na myšlence, že vygenerovaná DNA sekvence se částečně podobá DNA sekvencím z trénovací množiny. Takto experimentálně vygenerované DNA sekvence se mohou podobat DNA sekvencím vymřelých předků nebo příbuzných řádu pěvců.

Model byl schopný v části případů vygenerovat takové obrázky, které lze na pohled považovat za živočišný druh, ale je nutno konstatovat, že vygenerované obrázky často nešlo považovat za rekonstrukce živočichů. Úspěšnost vygenerování obstojného obrázku živočicha byla přibližně 10%.

Funkčnost modelu byla testována i na testovací množině DNA sekvencí živočichů několika řádů, které spadají pod třídu ptáků stejně jako řád pěvců. Úspěšnost vygenerování rekonstrukce, kterou bylo možné porovnávat s fotografií se pohybovala okolo 15%.

KLÍČOVÁ SLOVA

umělá neuronová síť, difúzní modely, generativní modely, DNA, GPT-2, Stable diffusion, cyklooxygenáza-1, ptáci, pěvci, vymřelé živočišné druhy, paleoart, transformátory, VAE, GAN, učení neuronových sítí

ABSTRACT

This work was focused on designing, learning and evaluating an artificial neural network for reconstructing extinct species. First, the main element of the proposed artificial neural network, i.e., the generative model, was selected. Given their excellent performance in the field of image generation, the class of diffusion models reasonably seemed to be the right choice. Specifically, the *Stable diffusion*[1] model was chosen.

One of the initial steps of the work was to create a training set for the proposed model. The animal images needed to be paired with some labels that could be used to identify the animal. For this purpose, the cytochrome c oxidase subunit I genes of the given animals were used.

Furthermore, the sequential transformer model GPT-2[2], which is learned on the training set of human natural language, was used. This model was used to encode the DNA sequences into a vector form in which the semantics and context between the different parts of the DNA sequence were captured.

The models would be very difficult to learn from scratch due to the large training set size required and the computational and time requirements. Thus, the GPT-2 model was only learned on the training set of DNA sequences of the passeriformes order, and the diffusion model itself was learned on pairs of images of these animals and DNA sequences encoded by the GPT-2 model.

To generate the images, the original DNA sequences that resembled the sequences from the training set were generated using GPT-2. The encoding of these sequences was then passed to the diffusion model, which generated the images itself. The method of generating new DNA sequences using the GPT-2 model is based on the idea that the generated DNA sequence partially resembles the DNA sequences from the training set. Such experimentally generated DNA sequences may resemble DNA sequences of extinct ancestors or relatives of the passeriformes order.

The model was in some cases able to generate images that could be considered as animal species , but it should be noted that often the generated images could not be considered as animal reconstructions. The success rate of generating a decent animal image was approximately 10%.

The functionality of the model was also tested on a test set of DNA sequences of animals of several orders that fall under the class of birds as well as the order of passeriformes. The success rate of generating a reconstruction that could be compared to a photograph was around 15%.

KEYWORDS

artificial neural network, diffusion models, generative models, DNA, GPT-2, Stable diffusion, cytochrome c oxidase subunit I, birds, songbirds, extinct species, paleoart, transformers, VAE, GAN, neural network learning

PEŠEK, David. *Umělá neuronová síť pro rekonstruování vymřelých druhů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2023, 61 s. Bakalářská práce. Vedoucí práce: doc. Ing. Václav Jirsík, CSc.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	David Pešek
VUT ID autora:	228732
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Umělá neuronová síť pro rekonstruování vymřelých druhů

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval konzultantovi bakalářské práce panu Ing. Dominiku Řičánkovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Také bych chtěl poděkovat rodině za podporu a pochopení v době práce na bakalářské práci.

Obsah

Seznam obrázků

Úvod

V posledních několika letech dochází k rozmachu pole působení umělých neuronových sítí. Hodně pozornosti získaly zejména generativní neuronové sítě, které dokáží generovat originální realistická díla jako je hudba, text a hlavně obrázky.

Tato práce je v první části zaměřena na generativní modely schopné vytvářet obrázky buď na základě vstupní textové pobídky nebo bez. Typický generativní model je DALL-E 2[7] od společnosti OpenAI, což je difúzní generativní model generující obrázky na základě textové pobídky v přirozeném jazyce.

Pro rekonstrukci vymřelých živočišných druhů je použit podobný postup s takovým rozdílem, že obrázky jsou generovány na základě DNA sekvencí.

Jako první krok je v práci vybrán konkrétní generativní model, který bude obrázky vytvářet. Generativní model je vybírán z architektur existujících generativních modelů. Následně tomuto generativnímu modelu bylo potřeba nějakým způsobem předat zakódované informace o živočiších, což je v práci řešeno pomocí transformátorového modelu naučeného na trénovací množině DNA sekvencí.

Pro tento úkol bylo potřeba najít způsob, jak a na jakých datech neuronovou síť správně naučit. Část práce je tedy vytvořit vhodnou trénovací množinu, na které je model trénován vytvářet obrazy vymřelých živočišných druhů.

Následně byl model naučen na nashromážděné trénovací množině za pomocí moderních trénovacích metod. Jakmile byl model naučen, bylo pomocí něho vygenerováno několik obrázků a funkčnost modelu ohodnocena. Také byl model testován na testovací množině a byla zjištěna úspěšnost vytvoření obstojného obrázku.

1 Výběr neuronové sítě pro rekonstrukci

1.1 GAN

GAN neboli Generative Adversarial Network je třída generativních neuronových sítí, která je postavena na myšlence hry s nulovým součtem, ve které jakýkoliv tah hráče 1 způsobí přírůstek v absolutní hodnotě stejně velký, jako je přírůstek hráče 2. Výsledek sečtení těchto dvou přírůstků je tedy nula.

Tyto sítě se obecně využívají pro generování fotorealistických obrázků, jako jsou fotografie zvířat, obličejů, panoramat atd. Dále jsou GAN také hojně používány při zpracovávání řeči či jazyka.

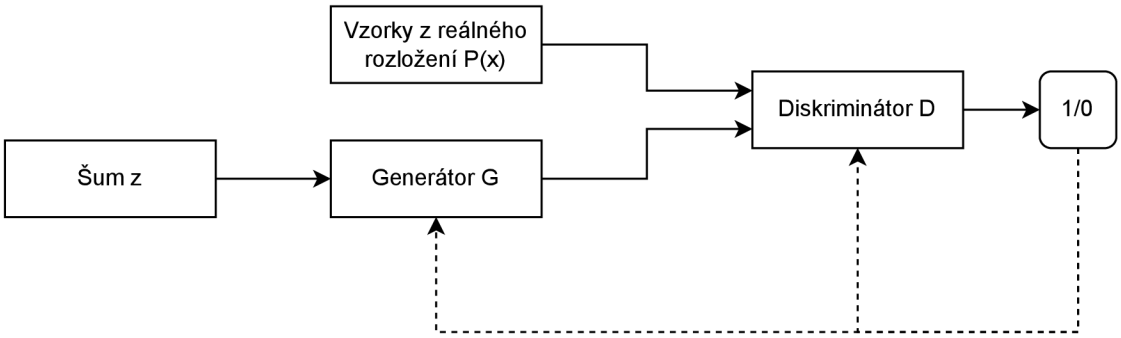
GAN se tedy skládá ze dvou hráčů. Těmito hráči jsou diskriminátor a generátor. Generátor má ve fázi učení za úkol odhadnout reálné rozdělení vstupních dat a dále toto rozdělení použít již pro generování autentických dat nerozpoznatelných od trénovacích dat. Naučení generátoru pro vytváření autentických dat se dosáhne právě pomocí diskriminátoru, který s generátorem hraje hru minimax, což je jedna z her s nulovým součtem. Diskriminátor je obvykle implementován jako binární klasifikátor rozhodující, zdali data z trénovací množiny jsou data reálná nebo data uměle vytvořená generátorem. Cyklus učení probíhá paralelně pro generátor i diskriminátor. To znamená, že ze začátku diskriminátor nedokáže spolehlivě poznat původ dat. Cíl této minimax hry je Nashova rovnováha, která nastává, když ani jeden z hráčů nemůže svým tahem dosáhnout vylepšení své situace. V GAN tento stav představuje případ, kdy generátor již dokázal najít rozdělení reálných dat z tréninkové sady, čili diskriminátor nedokáže rozpoznat reálná data od těch uměle vytvořených generátorem a již nedokáže konfigurovat svoje parametry pro dosažení lepší diskriminace.

Po dosažení Nashovy rovnováhy ztrácí smysl diskriminátor používat, protože je generátor již plně naučen. Diskriminátor se tedy odloží a dále se používá pro umělé vytvoření dat jen generátor.

1.1.1 Architektura GAN

Na obrázku 1.1 lze vidět obecnou GAN architekturu. Generátor G pomocí vzorků z odhadovaného rozdělení $p_z(\mathbf{z})$, kde $\mathbf{z}$ je náhodná proměnná, generuje vzorky $G(\mathbf{z})$.

Dále výstup z generátoru $G(\mathbf{z})$ je spolu se vzorky z reálného pravděpodobnostního rozdělení $p_{data}(\mathbf{x})$, kde $\mathbf{x}$ jsou reálná data, přiveden na vstup Diskriminátoru D, který má za úkol rozlišit reálné vzorky od uměle vytvořených. Tedy pokud bude na vstupu Diskriminátoru vzorek z Generátoru, tak na výstupu Diskriminátoru by měla být hodnota 0. Naopak když na vstupu Diskriminátoru bude vzorek $\mathbf{x}$ z reálného rozdělení, tak Diskriminátor odpoví hodnotou 1.



Obr. 1.1: Architektura GAN[3]

1.1.2 Učení GAN

Proces učení GAN je minmax hra. Lze ji popsat rovnicí:

$$min_G max_D\{f(D,G) = E_{x\ p_{data}(x)}[logD(x)] + E_{z\ p_z(z)}[log(1 - D(G(z))]\} \quad (1.1)$$

Kde:

$\mathbf{x}$	= reálná data z tréninkové množiny
$\mathbf{z}$	= náhodná data(šum) přivedený na vstup generátoru
$D(\mathbf{x})$	= odhad diskriminátoru, zdali data x jsou reálná
$G(\mathbf{z})$	= uměle vytvořená data generátorem na základě náhodných proměnných $\mathbf{z}$
$D(G(\mathbf{z}))$	= odhad diskriminátoru, zdali data G(z) jsou reálná
$E_{\mathbf{x}\ p_{data}(\mathbf{x})}$	= očekávaná hodnota na základě všech reálných dat
$E_{\mathbf{z}\ p_{\mathbf{z}}(\mathbf{z})}$	= očekávaná hodnota na základě všech náhodných proměnných $\boldsymbol{z}$ na vstupu generátoru

Při učení se tedy na tahu střídá Diskriminátor s Generátorem a nastavují svoje interní parametry. Generátor se snaží výše uvedenou funkci minimalizovat a diskriminátor maximalizovat.

Vzorec je odvozen z křížové entropie mezi reálnými a generovanými daty, která v podstatě hodnotí podobnost pravděpodobnostního rozdělení reálného a odhadovaného.

1.1.3 Shrnutí GAN

GANy výrazně přispěly k vývoji generativních modelů. Jednou z charakteristik je to, že vytváří data podobná datům trénovacím. Tím dokáží vytvářet velmi realistická díla. Tato podobnost k reálným datům ale vynáší na povrch jeden z problémů GANů, kterým je *"collapse mode problem"*. Tento problém nastává, když GAN není schopný

generovat dostatečně odlišná data. Například při generování realistických fotek se jednotlivé generace od sebe liší jen v barvě. Tento problém se může objevit, když generátor objeví určitý typ dat, který ošálí diskriminátor. Tím pádem již generátor nemusí upravovat svoje parametry a je doučen. Učení obvykle končí tedy nalezením lokálního minima.

1.2 VAE

Variational autoencoder(VAE) je generativní model schopný vytvářet vysoce realistická díla, jako jsou obrázky, zvuk nebo text. Autoenkodéry je také možno použít pro segmentaci obrázků.

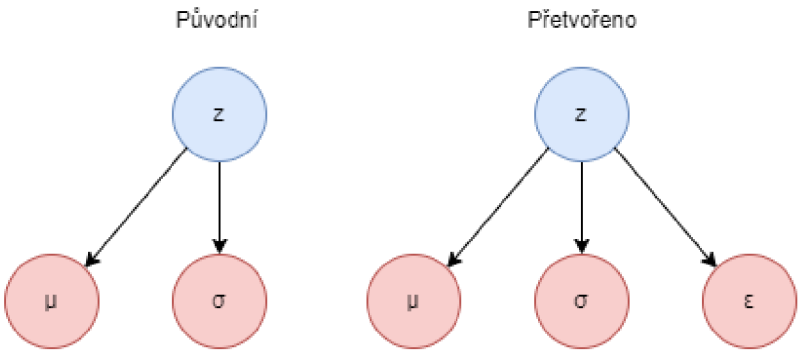
1.2.1 Enkodéry

Enkodéry jsou umělé neuronové sítě schopné zkomprimovat vstupní data na reprezentaci nižších rozměrů. Jsou tedy využívány pro snížení velikosti daného souboru. To je velmi užitečné například při posílání větších datových souborů přes síť. Enkodér bývá implementován jako několik konvolučních vrstev nebo jako plně propojená síť. Na výstupu této sítě je tzv. *bottleneck*, který reprezentuje vektor latentních proměnných, což jsou zmíněná komprimovaná data.

Pro rekonstrukci dat z vektoru latentních proměnných se využívá dekodér, který je stejně jako enkodér implementován buď jako několik konvolučních vrstev nebo plně propojená síť.

1.2.2 Architektura VAE

VAE vychází právě z výše zmíněných enkodérů. Na rozdíl od jednoduchých enkodérů, vstupní data jsou kódována na rozdělení hodnot, kterých mohou latentní proměnné nabývat. Dále ve fázi generování díla se extrahuje vzorek z tohoto rozdělení latentních proměnných a dekodér dílo vygeneruje. Pro učení sítě se využívá algoritmu zpětné propagace. Toto učení by ale nebylo možné, kdyby se *bottleneck* skládal ze vzorkovacího uzlu, který by obsahoval stochastickou operaci. Algoritmus zpětné propagace by přes tento uzel nemohl přenášet chybu dále do sítě. Proto je tento uzel rozdělen na více uzlů podle obrázku níže.



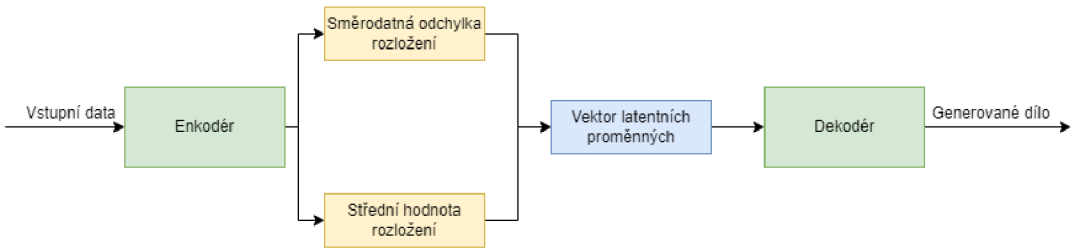
Obr. 1.2: Bottleneck[4]

Vzorec pro výpočet vektoru **z** je následující:

$$\mathbf{z} = \mu + \sigma \odot \epsilon \tag{1.2}$$

Kde:

$\mathbf{z}$ = vektor latentních proměnných
μ, σ = učené proměnné
ϵ = Stochastická část reprezentující Gaussovo rozdělení, ze kterého se vzorkuje. Tato část zůstává stejná, neprobíhá přes ní učení.



Obr. 1.3: Architektura VAE

1.2.3 Učení VAE

Při učení se vychází z předpokladu, že reálná data, která jsou použita pro učení, jsou považována za data vytvořená uměle. To znamená, že existuje takový proces, který tyto data vytvoří.

Řekněme, že máme množinu $X = \{x_i\}_{i=1}^{N}$ o N nezávislých veličin vytvořených neznámým procesem závislým na náhodné proměnné z. Tento proces probíhá tak, že z jsou vzorkovány z pravděpodobnostního rozdělení $p_\phi(z)$ a dále jsou tyto hodnoty použity pro generování dat x, které reprezentují nějaké tvořené dílo.

Parametry θ opravdového rozdělení $p_\theta(z)$ nelze zjistit, proto se uchylujeme k aproximování těchto parametrů $\phi \approx \theta$ pomocí dat x. Výsledkem je podmíněné pravděpodobnostní rozdělení $p_\phi(z|x)$. Paralelně při učení parametrů ϕ enkodéru jsou učeny i parametry dekodéru σ sloužící pro generování umělých dat x_u vzorkováním z rozdělení $p_\sigma(x|z)$.

1.2.4 Shrnutí VAE

Při generování originálních obrazů se vychází ze vzorku vektoru latentních proměnných $\mathbf{z} \sim p_\sigma(\mathbf{x}|\mathbf{z})$. To vede k problémům při generování obrazů. Část latentního prostoru může totiž postrádat jakýkoliv smysl vzhledem k velkém snížení rozměrů dat při snaze neztratit žádné informace o vstupních datech. VAE musí být tedy explicitně konfigurován tak, aby při trénování nedošlo k takovým chybám. Obecně jsou VAE velmi náročné na modifikaci a generují v porovnání s jinými generativními modely obrazy nižšího rozlišení.

1.3 Difúzní modely

Difúzní modely jsou *State of the art* generativní modely. Slouží ke generování dat podobných trénovacím datům. Fungují na principu postupného zašumění trénovacích dat a postupného naučení se obnovit data z tohoto šumu. Po naučení je tento model schopný z náhodného šumu vytvořit smysluplná data.

1.3.1 Dopředný proces difúze

Pomocí Markovova řetězce je postupně přidáván Gaussovský šum k trénovacím datům. Ke vzorku $\mathbf{x_0} \sim q(x)$ navzorkovaného z reálného rozdělení dat se tedy postupně přidává šum. Počet kroků přidávání šumu je dán rozvrhem rozptylu(*"Variance schedule"*) $\{\beta_t \in (0,1)\}_{t=1}^{T}$. Využití parametru β_t lze vidět ve vzorci níže. Se zvyšujícím se parametrem β_t se šum postupně přidává. Stojí za zmínku, že při $T \to \infty$ je $\mathbf{x_T}$ rovno izotropnímu Gaussovu rozdělení.

$$q(\mathbf{x_t} \mid \mathbf{x_{t-1}}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \tag{1.3}$$

$\mathbf{x_t}$ = zašuměná data v kroku t
β_t = proměnná normálního rozdělení

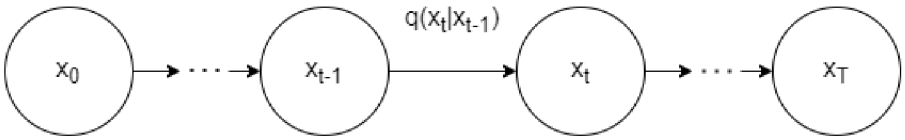
Vzhledem k tomu, že je znám rozvrh zašumování vstupních dat, tak je možno vypočítat x_t v libovolném kroku dopředné difúze. Ovšem nebylo by takto možno

vzorkovat bez přetvoření daného uzlu pomocí triku reparametrizace, kde rozdělíme stochastickou operaci vzorkování x_t na operace nestochastické a stochastickou operaci vzorkování z normálního rozdělení.

$$\begin{aligned} \mathbf{x_t} &= \sqrt{\alpha_t}\mathbf{x_{t-1}} + \sqrt{1-\alpha_t}\epsilon_{t-1} \\ &= \sqrt{\bar{\alpha}_t}\mathbf{x_0} + \sqrt{1-\bar{\alpha}_t}\epsilon \\ q(\mathbf{x_t} \mid \mathbf{x_0}) &= \mathcal{N}(\mathbf{x_t}; \sqrt{\bar{\alpha}_t}\mathbf{x_0}, (1-\bar{\alpha}_t)\mathbf{I}) \end{aligned} \tag{1.4}$$

$\mathbf{x_t}$ = zašuměná data v kroku t
α_t = $(1-\beta_t)$
$\bar{\alpha}_t$ = $\Pi_{i=1}^{t}\alpha_i$
ϵ_{t-1} = vzorek z Gaussova rozdělení $\epsilon_{t-1} \sim \mathcal{N}(0, \mathbf{I})$
ϵ = $\Pi_{i=1}^{t}\epsilon_{i-1}$ sloučené Gaussovy křivky

Možnost výpočtu libovolného kroku zašumování se bude dále hodit při reverzním procesu difúze.



Obr. 1.4: Dopředný proces difúze

1.3.2 Reverzní proces difúze

V tuto chvíli máme na vstupu vzorek z normálního rozdělení $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Z tohoto vzorku bychom teoreticky mohli reverzováním dopředného procesu difúze vytvořit reálná data před zašuměním pomocí vzorkování z $q(x_{t-1} \mid x_t)$ od t = T až po t = 1.

Tento postup by byl ale velmi výpočetně náročný, a proto se uchylujeme k naučení modelu p_θ jako aproximaci podmíněných pravděpodobností $q(x_{t-1} \mid x_t)$.

$$\begin{aligned} p_\theta(\mathbf{x_{0:T}}) &= p(\mathbf{x_T})\Pi_{t=1}^{T} p_\theta(\mathbf{x_{t-1}} \mid \mathbf{x_t}) \\ p_\theta(\mathbf{x_{t-1}} \mid \mathbf{x_t}) &= \mathcal{N}(x_{t-1}; \mu_\theta(\mathbf{x_t}, \mathbf{t}), \Sigma_\theta(\mathbf{x_t}, t)) \end{aligned} \tag{1.5}$$

Reverzace procesu dopředné difúze použitého pro zašumění obrazu:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \tag{1.6}$$

Šum přidáván k obrazu v každém kroku dopředné difúze je parametrizován na střední hodnotě $\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0)$ a rozptylu $\tilde{\beta}_t$. Tyto parametry se v každém kroku mění.

Rozptyl $\tilde{\beta}_t$ je deterministická proměnná závislá na rozvrhu rozptylu použitém při dopředném procesu difúze. Lze ji tedy vypočítat z rozptylu v daném kroku t. Nutno držet v paměti, že $\alpha_t = 1 - \beta_t$ a $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$

$$\tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \tag{1.7}$$

Střední hodnota datových bodů závisí stejně jako rozptyl na rozvrhu rozptylu. Navíc ale musí záviset na datech před zašuměním x_0 a na datech x_t v kroku zašumění t.

$$\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}\mathbf{x}_0 \tag{1.8}$$

Vzhledem k tomu, že dokážeme zjistit data $\mathbf{x}_t$ v každém kroku t (vzorec č.1.4), můžeme zjistit data x_0 v kroku $t = 0$ a rovnici přepsat na tvar:

$$\begin{aligned} \tilde{\boldsymbol{\mu}}_t &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}_t) \\ &= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \end{aligned} \tag{1.9}$$

Učení

Při učení hledáme pomocí nástroje *Variational lower bound* takové parametry U-NET aby aproximace rozdělení pravděpodobnosti byla co nejvíce podobná reálnému rozdělení.

Funkci reprezentující správnost aproximovaného pravděpodobnostího rozdělení reverzního procesu difúze lze zapsat jako řetězec po sobě jdoucích VLB každého aproximovaného přechodu Markovova řetězce.

$$L_{\text{VLB}} = L_T + L_{T-1} + \cdots + L_0$$
$$L_{\text{VLB}} = L_T + L_t + L_0$$

Kde L_T reprezentuje VLB dopředného procesu zašumování obrazu. Vzhledem k tomu, že rozvrh rozptylu β_t je fixovaný, tak je L_T konstanta, kterou můžeme při učení zanedbat.

L_t reprezentuje VLB reverzního procesu pro $t \in (1, T-1)$. Při učení chceme aproximovat reálné rozdělení tak ,jak je popsáno v sekci 1.3.2 *"Reparametrizace L_t pro učení"*.

Reparametrizace L_t pro učení

Při generování obrazů již nebude nezašuměný obraz $\mathbf{x_0}$ k dispozici. Nebudeme tedy znát ani šum v obrazu. Proto se uchylujeme k naučení konvoluční neuronové sítě U-NET. Tato síť je vhodná zejména díky tomu, že na vstupu i výstupu této sítě jsou data stejné dimenzionality.

U-NET učíme, aby aproximovala $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$ k reálnému podmíněnému rozdělení pravděpodobnosti
$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$. Podobnost těchto dvou rozdělení pravděpodobnosti lze zjisti pomocí *Kullback–Leibler divergence*(KL divergence) reálného rozdělení a toho aproximovaného. $L_t = D_{\mathrm{KL}}(q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}))$ pro $1 \leq t \leq T-1$.

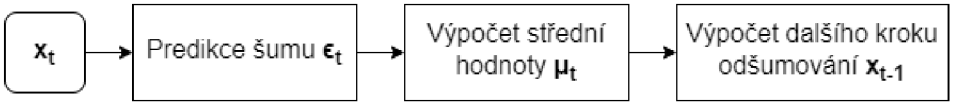
Vzhledem k tomu, že je rozptyl normálního rozdělení ve funkci p_θ daný rozvrhem rozptylu dopředného procesu, zbývá nám zjistit jen střední hodnotu $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$. Dále tedy můžeme ztrátovou funkci modifikovat na minimalizaci rozdílu mezi reálnou střední hodnotou $\tilde{\boldsymbol{\mu}}_t$ a aproximovanou střední hodnotou $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$.

$$L_t \propto \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2 \tag{1.10}$$

Rozdělení pravděpodobnosti je tedy zjišťováno tak, že učíme funkci $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ predikovat reálnou střední hodnotu dopředného procesu $\tilde{\boldsymbol{\mu}}_t$. Při učení máme k dispozici obraz $\mathbf{x_t}$ a jediná neznámá je šum $\boldsymbol{\epsilon}_t$ obrazu $\mathbf{x}_t$ v kroku t. Můžeme tedy reparametrizovat funkci normálního rozdělení tak, aby byla U-NET naučena na predikci šumu $\boldsymbol{\epsilon}_t$ obrazu $\mathbf{x}_t$ v kroku t.

$$\begin{aligned} \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) &= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) \\ \mathbf{x}_{t-1} &= \mathcal{N}(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \end{aligned} \tag{1.11}$$

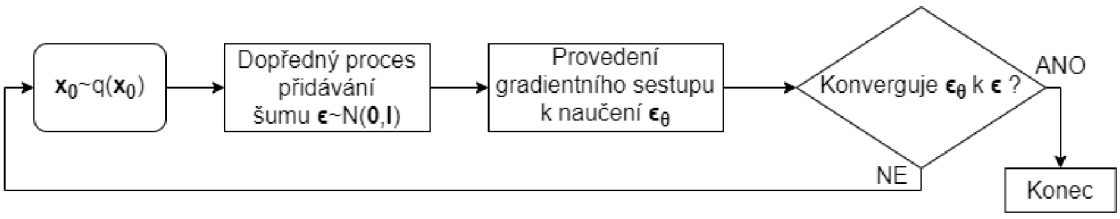
V reverzním procesu difúze tedy každý krok odšumování probíhá jako vzorkování z naučeného rozdělení pravděpodobnosti $\mathbf{x_{t-1}} \sim p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$.



Obr. 1.5: Jeden krok odšumování obrazu při generování

Uvedením $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ jako učené funkce je ztrátová funkce definována takto:

$$\begin{aligned} L_t &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2\right] \\ &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\|^2\right] \end{aligned} \tag{1.12}$$



Obr. 1.6: Postup při učení U-NET sítě difúzního modelu

1.4 Kullback–Leibler divergence

KL divergence je statistický nástroj pro měření rozdílnosti, statistické vzdálenosti dvou rozdělení pravděpodobnosti. Obvykle se používá pro porovnání nějakého referenčního rozdělení P a aproximací tohoto rozdělení Q.

V ML se obvykle používá ve výpočtu ztrátové funkce pro ohodnocení aproximace Q při učení generativních modelů.

$$D_{KL}(P\|Q) = \sum P(x) \log\left(\frac{P(x)}{Q(x)}\right) \tag{1.13}$$

1.5 Variational lower bound

V ML je variational lower bound (VLB) používán pro aproximování složitého pravděpodobnostního rozdělení na méně komplexní rozdělení. Úloha tohoto nástroje je najít spodní mez logaritmické pravděpodobnosti dat, což jde optimalizovat tak, aby byla nalezena nejlepší aproximace reálného pravděpodobnostního rozdělení.

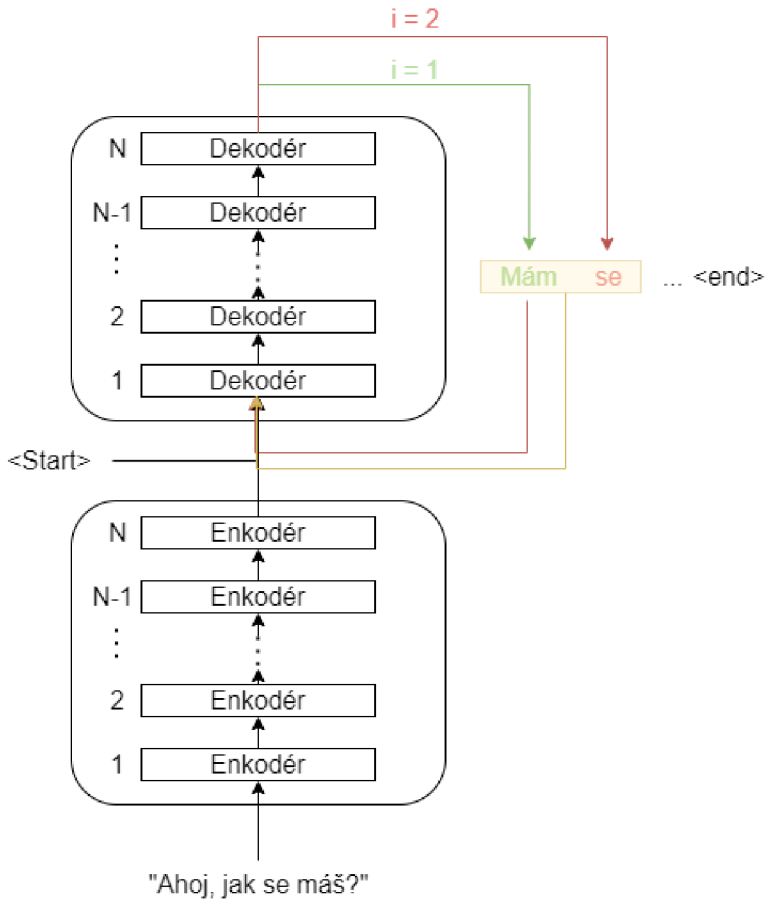
$$L = \log p(X) - KL\left[q(Z)\|p(Z|X)\right] \tag{1.14}$$

Jak lze vidět ve vzorci č.1.14 je definována jako rozdíl KL divergence reálného a aproximovaného rozdělení pravděpodobnosti a logaritmické pravděpodobnosti reálného rozdělení. KL divergence je vždy ≥ 0, odečtením od $\log p(X)$ tedy získáme spodní mez logaritmické pravděpodobnosti dat.

1.6 Jazykové transformátory

Modely neuronových sítí zmíněných výše jsou generativní modely, které dokáží generovat originální obrázky podobné těm z trénovací množiny. To ale nestačí, když potřebujeme generovat obrázky na základě textové pobídky.

Pro nasměrování generovacího modelu do oblasti dané textovou pobídkou přicházejí na scénu jazykové transformátory, které transformují text z přirozeného jazyka do latentní vektorové reprezentace, ze které dále generativní model vytvoří odpovídající obraz.



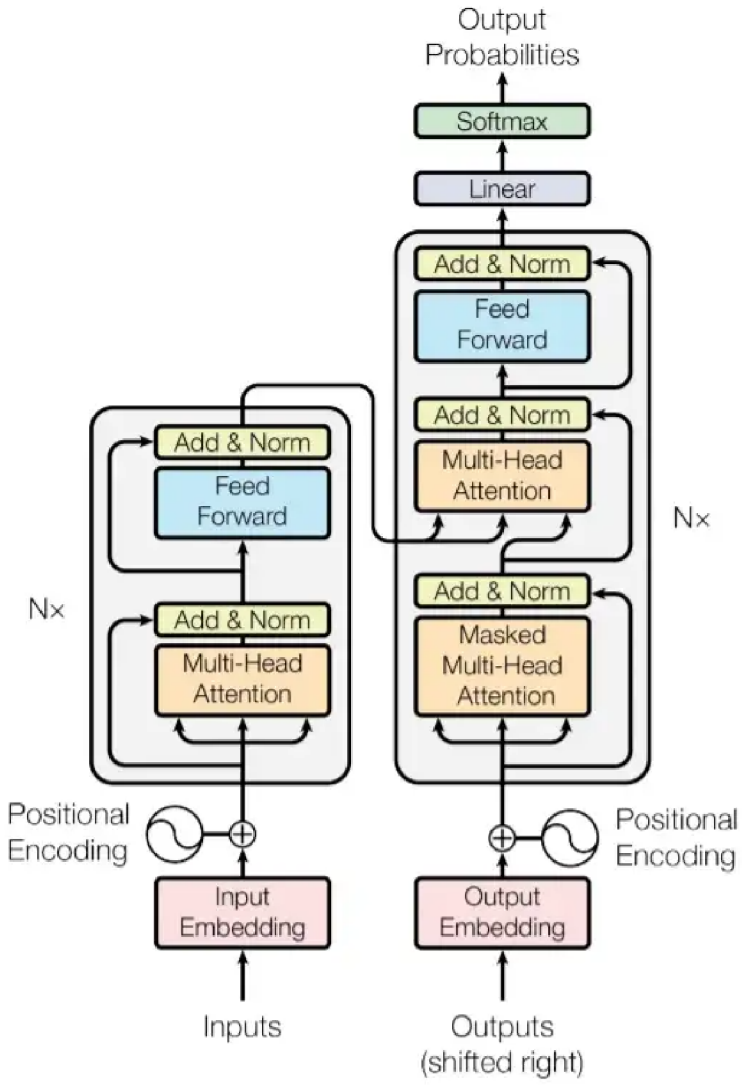
Obr. 1.7: Ukázka funkce generování textové sekvence ze vstupní sekvence

1.6.1 Architektura transformátorů

Transformátory se často využívají pro generování sekvence slov na základě vstupní sekvence slov. Vstupní sekvence slov je předložena enkodéru, který ji zakóduje do abstraktních vektorů spojitých hodnot. Tyto vektory jsou dále dekodérem využity

pro generování výstupní sekvence slov. Transformátory fungují sekvenčně. To znamená, že slova zpracovávaná enkodérem jsou kódována postupně, tak jak jdou za sebou ve větě. Dekodéry vytváří výstupní sekvenci také postupně, slovo po slově.

Je běžné, že transformátory obsahují několik po sobě jdoucích enkodérů a dekodérů. To zajišťuje zachycení hlubších vazeb mezi slovy.



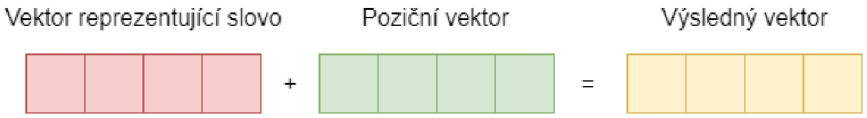
Obr. 1.8: Architetura transformátorového modelu[5]

Self attention

Klíčová součást jazykových transformátorů je vrstva self attention, která definuje vazbu slov v dané sekvenci slov. Při komunikaci v přirozeném lidském jazyce se klade velký důraz na kontext. Mějme například větu: *"Pes je nejlepší přítel člověka."*

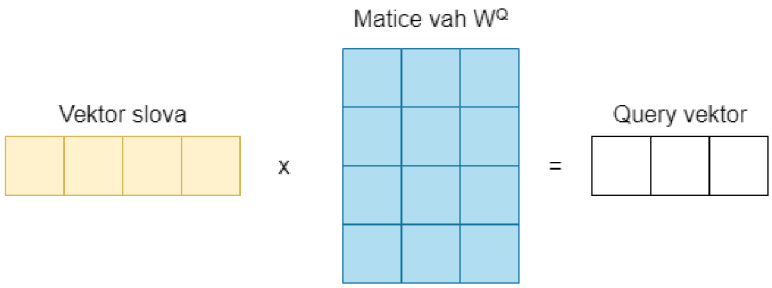
Jakožto člověk vím, že slovo přítel referuje na slovo pes. Self attention vrstva tedy pomáhá transformátorům tyto vazby extrahovat.

Aby tato vrstva a celý transformátor mohl se slovy pracovat, musí se nejdříve přetransformovat do vektorů. V současnosti existující volně přístupné jazykové transformátory jako je GPT-2 mají předem naučeny vektorové reprezentace velkého množství slov. Například GPT-2 je naučen na více než 50 000 slov v angličtině. Slova v přirozeném jazyce mají také různý význam v závislosti na tom, na jaké pozici ve větě se nachází, proto je ještě k těmto vektorům přičteno poziční kódování.



Obr. 1.9: První vektorová reprezentace slova

Takto vytvořený vektor již putuje do self attention vrstvy. Každý vektor zpracovávaný v této vrstvě je dále přetvořen do tří dalších vektorů, a to key, query a value. Při zjišťování vazeb slova *"člověka"* ve větě *"Pes je nejlepší přítel člověka."* se vytvoří query vektor tohoto slova, je to reprezentace tohoto slova pro srovnávání s dalšími slovy ve větě. Key vektory ostatních slov ve větě jsou použity k porovnávání s query vektorem. Ve vektoru value je umístěna reálná reprezentace daného slova. Tyto vektory jsou vytvořeny maticovým násobením vstupní vektorové reprezentace slova maticemi vah W^Q, W^K a W^V.



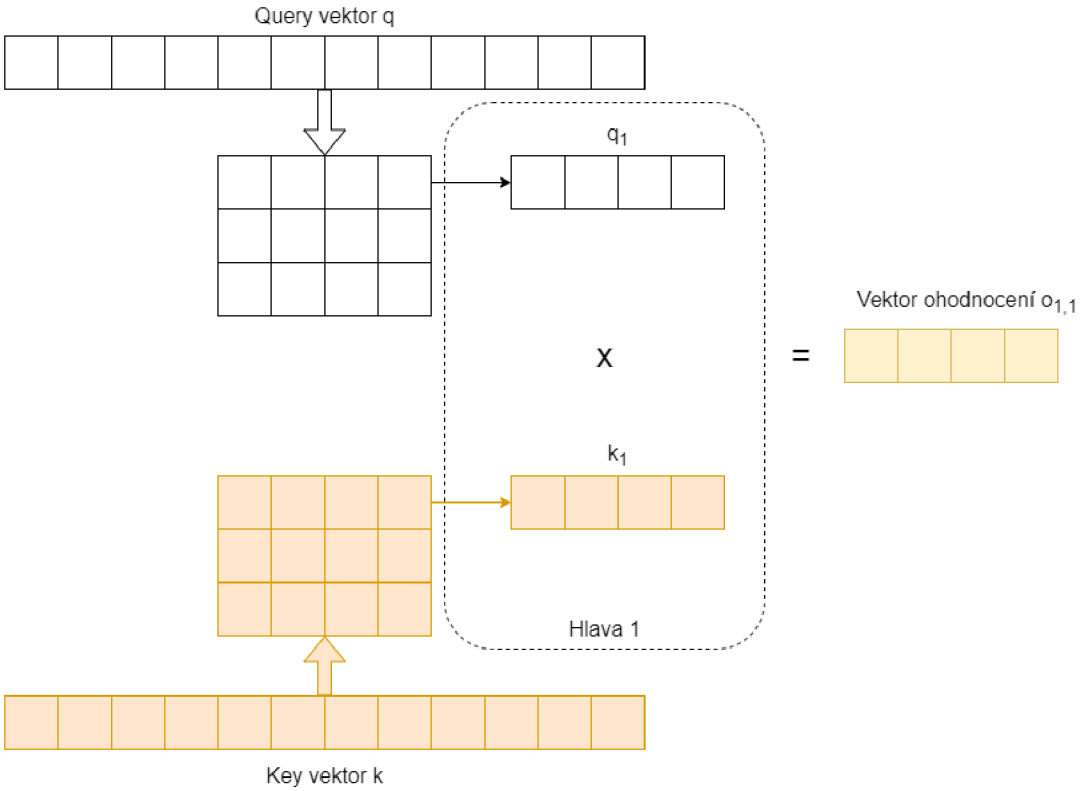
Obr. 1.10: Query vektor slova

Dále je proveden skalární součin query vektoru s key vektory zbylých slov ve větě, čímž po aplikování funkce softmax(konvertování vektoru čísel na vektor jejich pravděpodobností) získáme hodnocení síly vazby mezi slovem, který je reprezentován query vektorem a každým slovem reprezentovaným key vektorem.

Vzniklé vektory hodnocení jsou dále vynásobeny jejich value vektory, čímž se nastaví váha tohoto slova vůči řešenému slovu. Všechny tyto vzniklé vektory jsou dále sečteny a přivedeny na vstup dopředné neuronové sítě, která převede vektor zpět do rozměrů na vstupu self attention vrstvy.

Multi-headed attention

Multi headed self attention je rozdělení procesu self attention do několika paralelně běžících subprocesů. Po vytvoření query, key a value matic jsou tyto matice rozděleny na tolik částí, kolik hlav model má. Dále je proces podobný jako u běžného self attention procesu. Jsou vypočítány dílčí vektory ohodnocení.



Obr. 1.11: Ukázka části výpočtu ohodnocení jedné hlavy

Na obrázku 1.11 je zobrazen výpočet dílčího ohodnocení hlavou č.1 s jediným key vektorem. V realitě jsou v jedné hlavě vypočteny ohodnocení se všemi key vektory. Ve všech hlavách dochází k vypočtení dílčího ohodnocení jedné části query vektoru se stejnými částmi key vektorů všech ostatních slov. Tyto dílčí vektory ohodnocení ze všech hlav jsou spojeny do jednoho vektoru a s maticí vah, které se nastaví při učení je vynásoben do podoby vektoru, který je pro dopřednou neuronovou síť vhodný.

Masked self attention

Při učení transformátorů je k dispozici celá výstupní sekvence slov. Proto, aby dekodér "nepodváděl"je nutno slova generovaná po slově, které je právě na vstupu dekodéru nějakým způsobem pro dekodér zneviditelnit.

Masked self attention blok je varianta self attention, kdy v kroku vytváření vektorů hodnocení dekodérem je ignorováno hodnocení query vektoru s key vektory slov, která budou generována v budoucnosti. Toho je dosáhnuto přičtením maskovací matice k matici ohodnocení předtím, než projde funkcí softmax.

2 Metoda

2.1 Trénovací množina

Pro rekonstrukci vymřelých živočišných druhů byly použity páry DNA sekvencí a obrázků odpovídajících živočichů. Jako popisek zvířete se nabízelo místo DNA sekvencí použít například kostní pozůstatky těchto živočichů. DNA sekvence byly ale jako popisek použity kvůli volnému přístupu k těmto datům, na první pohled velké databázi národní lékařské knihovny Spojených států amerických a existující python knihovně a API zjednodušující automatizované stahování DNA sekvencí.

DNA

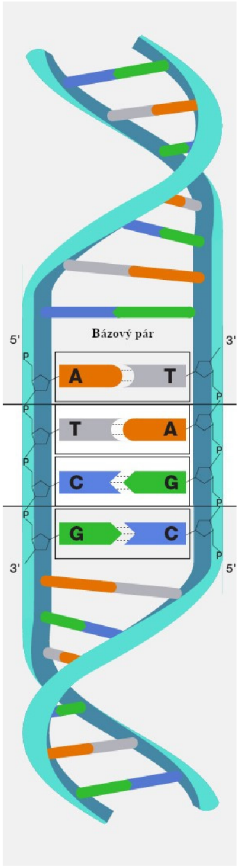
Přístup k rekonstrukci vymřelých živočišných druhů je založený na faktu, že DNA sekvence určité skupiny žijících živočichů se částečně podobá DNA sekvenci společného předka těchto živočichů nebo DNA vymřelého příbuzného. Takže pokud vygenerujeme DNA sekvenci podobnou DNA sekvencím živočišných druhů v dané třídě živočichů, existuje šance, že vygenerujeme DNA sekvenci předka těchto živočichů.

Konkrétně byl vybrán řád pěvci z třídy ptáci. Při výběru byla snaha o kompromis mezi počtem dostupných DNA sekvencí pro členy skupiny živočichů a podobností mezi členy skupiny. Cíl byl tedy vybrat skupinu živočichů takovou, aby trénovací množina nebyla příliš malá a zároveň, aby šlo o skupinu živočichů sobě podobných, například o řád živočichů a ne celou třídu živočichů.

DNA sekvence v sobě ukrývají genetickou informaci specifickou pro jedince, ale i pro určitý živočišný druh. Proto je možno DNA použít k identifikaci druhu živočicha. Samotné DNA sekvence jsou posloupnosti nukleotidů, jejichž součástí jsou posloupnosti čtyř různých nukleových bází. Tyto báze jsou adenin, thymin, guanin a cytosin, které jsou značeny jako A, T, G a C. Dále tyto báze se na sebe vážou a utváří bázové páry, kde adenin se páruje s thyminem a cytosin s guaninem.

Prvotní nápad byl pro učení použít celé DNA sekvence živočichů, ale tyto sekvence jsou velmi dlouhé a proto by nebylo možné je použít jako vstup pro použité UNS. Z tohoto důvodu byla použita pouze část genomu živočichů, a to cyklooxygenáza-1 (COX-1). Jedná se o enzym, který je součástí mitochondriální DNA všech živočichů patřících do domény eukaryota[8]. Tento gen je obvykle používán pro identifikaci organismů pomocí metody zvané *barcoding*[9] a je tedy pro každý živočišný druh unikátní. Tímto byla výrazně snížena délka použitých DNA sekvencí. A to přibližně z jedné miliardy bázových párů až na asi 1500 bázových párů.

Trénovací množina tedy obsahuje 23 DNA sekvencí zástupců řádu pěvci. DNA sekvence jsou jak už bylo zmíněno pouze částečné. Konkrétně COX-1 geny těchto



Obr. 2.1: Vizualizace DNA šroubovice s bázovými páry[6]

živočichů.

Obrázky

K DNA sekvencím byly následně přiřazeny 3 různé fotografie příslušných živočichů. Samotné fotografie byly získány z platformy flickr, na které uživatelé sdílí různé fotografie či obrázky. Stahování obrázků z této platformy zjednodušilo použití volně přístupné flickr API v programovacím jazyce python. Pro učení neuronové sítě byly fotografie z tréninkové množiny transformovány na stejné rozměry 512x512 pixelů.

Obr. 2.2: Ukázka několika obrázků z trénovací množiny

2.2 Zakódování a generování DNA

Postup rekonstruování vymřelých druhů představený v této práci je takový, že pro vygenerování obrázků těchto druhů nejdříve vygenerujeme pomocí určité neuronové sítě (transformátorový model) originální DNA sekvenci, kterou dále předáme generativnímu difúznímu modelu, který z ní vygeneruje obrázek tohoto živočicha.

Jelikož DNA jsou sekvenční data, je příhodné použít na generování a zároveň na zakódování DNA sekvencí jazykový transformátor. Jazykové transformátory jsou umělé neuronové sítě učené na sekvencích slov psaného lidského jazyka. Obvykle jsou používány pro překlad mezi lidskými jazyky, generování textu nebo jako všestranný asistent jako například chatGPT od společnosti OpenAI.

Je nutno podotknout, že transformátory pro práci s DNA sekvencemi již použili autoři transformátorového modelu DNABert[10]. Autoři použili transformátor pro různé predikční úlohy v oblasti DNA.

Učit vlastní model založený na transformátorech by bylo velmi náročné z hlediska času, nákladů a kvůli potřebné velikosti trénovací množiny. Například transformátorový model GPT-2[2] od společnosti OpenAI byl učen na trénovací množině textu o velikosti 40 GB. Naštěstí existují volně přístupné předučené transformátorové modely, které lze použít pro vlastní potřeby.

Již zmíněný GPT-2 model byl sice naučen na textu lidského jazyka, ale pomocí metody doladění(v angličtině *fine-tuning*) je možno tento předučený model použít pro vlastní specifické účely.

3 GPT-2

K doladění byl vybrán model GPT-2 od společnosti OpenAI, díky jeho schopnostem, které jsou na úrovni moderního volně dostupného transformátorového modelu. Další důvod použití tohoto modelu je také dobrá dostupnost dokumentace a fakt, že jeho kód je volně přístupný.

3.1 Slovník DNA

GPT-2 je jazykový model využívající dekodérů z transformátoru, který byl představen v práci *Attention is all you need*[5]. Těchto dekodérů je v modelu 12 vrstev zapojených za sebou. Funkce dekodérů je popsána v sekci 3.3

Samotná architektura základního GPT-2 vypadá následovně. První operace se vstupní větou je tokenizace. Každé slovo ve vstupní větě je převedeno na token podle slovníku daného modelu. Tokenem je myšlena číselná reprezentace slova. To znamená, že kdybychom převáděli například větu *"Pes je nejlepší přítel člověka."*, tak po tokenizaci by věta vypadala přibližně takto: 5 3 58 94 23.

Vzhledem k tomu, že se v práci nezabýváme lidským jaykem, ale DNA sekvencemi, tak bylo nutno vytvořit vlastní slovník. Nabízelo se vytvořit slovník obsahující pouze čtyři slova, tedy nukleové báze A, T, G a C. Tento přístup by ale znamenal příliš dlouhou vstupní sekvenci, protože maximální délka vstupní sekvence pro GPT-2 je 1024 a délky našich vstupních DNA sekvencí jsou přibližně 1500 nukleových bází.

Tento problém byl vyřešen rozdělením vstupní DNA sekvence na úseky dlouhé šest nukleových bází. Tímto byla délka vstupních sekvencí šestkrát snížena z 1500 na přibližně 250 slov. Slovník, který byl tedy použit k tokenizaci obsahuje $4^6 = 4096$ slov. Je možné, že tímto bylo ztraceno nějaké množství informací uložených v DNA sekvenci kvůli spojení několika nukleových bází do jednoho tokenu, ale jak bylo řečeno, vstupní sekvence by jinak byla příliš dlouhá. Bylo by také možno DNA sekvence rozdělit například na úseky dlouhé 2 nukleové báze, ale větší délka sekvence by zase vedla k větší výpočetní náročnosti při učení a generování nových sekvencí.

3.2 Embedding vrstvy GPT-2

Následně po tokenizaci slov je věta vložena na vstup tzv. *token embedding* vrstvy. V této vrstvě je každý token zakódován do 768 rozměrného vektoru. Každému slovu, tedy tokenu, odpovídá právě jedno vektorové zakódování. GPT-2 je sekvenční model, který zachycuje sémantiku vět a vztah mezi slovy na různých místech ve větě. Proto je potřeba modelu nějakým způsobem předat i informaci o pozici každého slova ve větě. K tomu je používáno poziční kódování, kde podobně jako v předchozí *token*

embedding vrstvě je podle pozice slova ve větě vypočten vektor o 768 rozměrech, který je následně přičten k předchozímu vektoru.

3.3 Dekodér

Dále jsou tyto vektory slov sekvenčně předávány sekvenci dvanácti dekodérů. Každý dekodér se skládá z *self attention* bloku, jehož funkce je vysvětlena v sekci 1.6 a z dopředné neuronové síti. Tyto dopředné neuronové sítě hrají klíčovou roli v transformátorech. Na vstupní vrstvu této dopředné sítě je přiveden výstup z *self attention* bloku, tedy vektorové reprezentace slov obohacené o informaci, kterým částem věty má dávat větší váhu.

Skryté vrstvy zachycují skryté korelace mezi daty na jejich vstupu a dále výstupní vrstva dopředné neuronové sítě předává informace dalšímu dekodéru. Díky tomu, že dekodérů je v transformátoru více, je transformátor schopný zachytit složité vzorce a závislosti mezi vstupními daty. Každý dekodér přetváří reprezentace z předchozího dekodéru, což má za důsledek hluboké kontextově informované porozumění vstupního textu.

3.3.1 Zakódované DNA sekvence

Právě hodnoty z posledního dekodéru, konkrétně z výstupní vrstvy dopředné neuronové sítě byly použity jako kódování DNA sekvence. To znamená, že DNA sekvence dlouhá 250 tokenů byla zakódována do matice o rozměrech 250x768.

3.4 Lineární vrstva

Na vstupu této vrstvy je napojen výstup z posledního dekodéru. Tato vrstva se učí lineární závislost jejího výstupu na jeho vstupu. Vrstva je například používána při generování nových dat nebo doplňování chybějících částí dat. Výstupní rozměry odpovídají velikosti slovníku. Mějme tedy například sekvenci slov *"Pes je nejlepší přítel"*. Tato slova jsou vložena na vstup transformátorového modelu a každý neuron ve výstupní vrstvě bude na výstupu odpovídat tím větší hodnotou, čím je větší pravděpodobnost výskytu příslušného tokenu (slova) jako pokračování vstupní sekvence.

Hodnoty z výstupní vrstvy jsou dále softmax funkcí převedeny na pravděpodobnostní rozdělení tokenů.

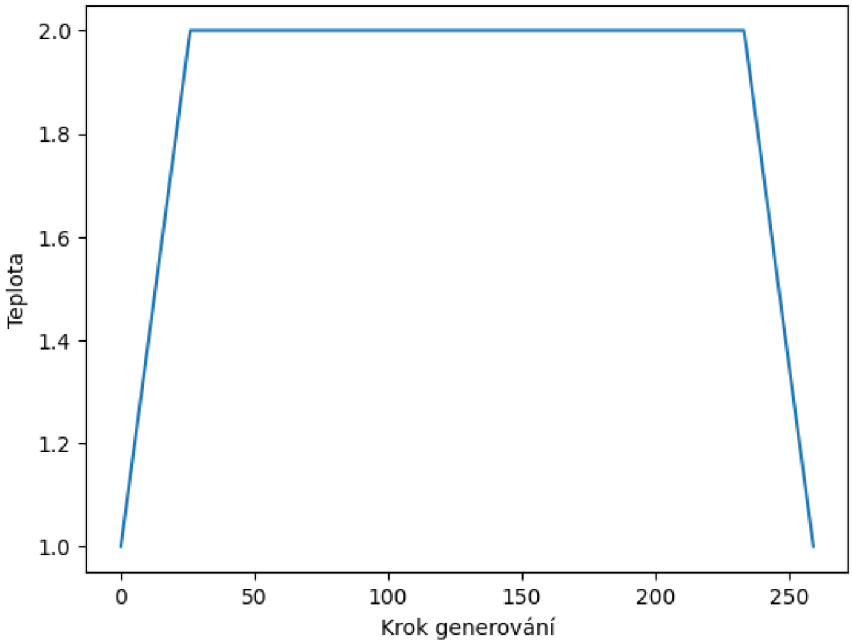
3.4.1 Generování DNA sekvencí

Proces generování nových DNA sekvencí tedy probíhá tak, že je na vstup GPT-2 přiveden *start* token. Model odpoví pravděpodobnostním rozdělením tokenů. Z tohoto rozdělení je dále vzorkováno metodou top-p, což znamená, že se další možné tokeny seřadí sestupně podle jejich pravděpodobnosti a další token se navzorkuje z podmnožiny tokenů, jejichž kumulativní pravděpodobnost nepřekračuje hodnotu p. Při generování DNA sekvencí v této práci byla použita hodnota $p = 60\%$.

Tento vzorek představující první token DNA sekvence, je připojen ke *start* tokenu a následně je celá sekvence znovu přivedena na vstup modelu. Tento algoritmus běží dokud není navzorkován *end* token značící konec DNA sekvence.

Pro lepší rozmanitost generovaných sekvencí byl postup generování doplněn ještě o jeden mezikrok vzorkování s teplotou[11]. Před tím než je aplikována funkce softmax na hodnoty z výstupní lineární vrstvy, jsou tyto hodnoty vyděleny hodnotou teploty. Hodnota teploty větší než 1 způsobí větší nejistotu modelu k odhadu dalšího tokenu, tedy tokeny, které měli vyšší šanci na navzorkování mají nyní šanci menší a tokenům s nižší šancí se šance naopak zvedne.

Teplota se řídí rozvrhem teploty, který udává jaká hodnota teploty je v daném kroku generování DNA sekvence nastavena.



Obr. 3.1: Rozvrh teploty při generování originálních DNA sekvencí

Na obrázku 3.1 lze vidět, že nejdříve se teplota se zvyšujícím se krokem lineárně zvyšuje. Rozvrh byl takto nastaven, aby hned ze začátku model nenavzorkoval token s velmi malou pravděpodobností a tím se nedostal do oblasti, která nebyla při učení

dostatečně naučena. Ke konci je znovu snižována, aby model generoval ty vzorky co jsou co nejvíce pravděpodobné.

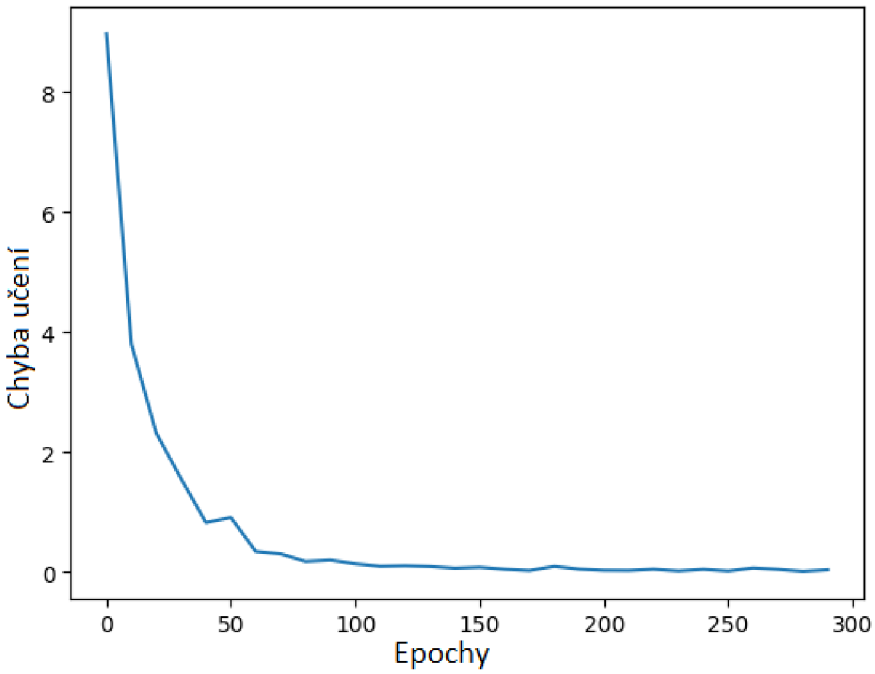
3.5 Doladění GPT-2

Jak bylo již zmíněno, pro doučení GPT-2 bylo použito 27 DNA sekvencí genu cyklooxygenázy-1 zástupců řádu pěvců. Podle postupu v sekci 3.1 byly DNA sekvence převedeny na tokeny. Nutno ještě dodat, že každé DNA sekvenci byl na počátek přidán *start* token, aby při generování bylo možno předat modelu pouze tento token a model byl schopný vygenerovat novou DNA sekvenci. Podobně jako *start* token byl na konec každé DNA sekvence přidán *end* token, aby se model naučil správnou délku DNA sekvence.

Při samotném učení byly na vstup GPT-2 modelu předávány celé DNA sekvence cyklooxygenázy-1. Model postupně odhaduje tokeny celé sekvence. To znamená, že nejdříve je přiveden na vstup modelu první token a je odhadnut následující token. Dále je na vstup přiveden další známý token a je odhadován token na další pozici. Takto postupuje až na konec vstupní sekvence.

Chyba učení je vypočítávána jako průměrná chyba křížové entropie mezi aktuálním výstupem lineární vrstvy po aplikování softmax funkce, tedy rozdělení pravděpodobnosti předvídaných tokenů a tokenem, který reálně je na dalším místě v DNA sekvenci.

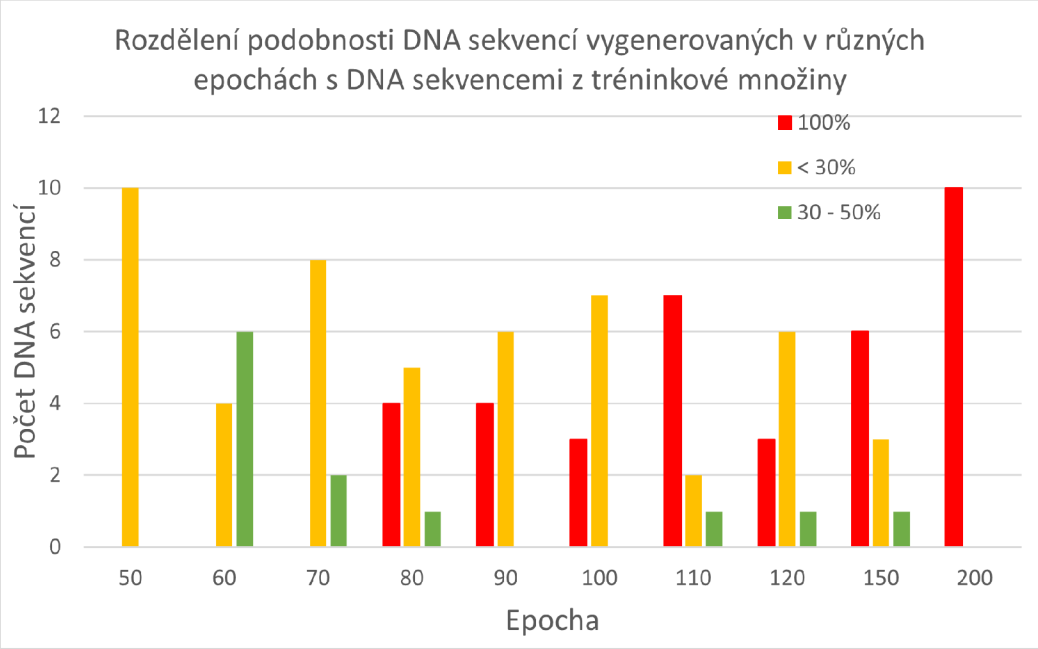
Učení probíhalo v 200 epochách, kde v každé epoše byly váhy neuronové sítě upraveny po vypočítání chyby každého z 27 pěvců v trénovací množině. Při testování byl ale odhalen problém. Model po 200 epochách již nebyl schopný generovat originální data a generoval pouze DNA sekvence totožné těm, na kterých byl učen.



Obr. 3.2: Závislost chyby učení GPT-2 na epoše

Byly tedy testovány instance modelu z různých epoch učení. Jako metriku správnosti modelu byla použita jak chyba učení, která nesměla být příliš velká, tak i procento vygenerovaných originálních DNA sekvencí, které byly DNA sekvencím totožné v 30 až 50 %. Tato míra totožnosti byla zjištěna porovnáváním DNA sekvencí v trénovací množině, kde se totožnost mezi jednotlivými DNA sekvencemi pohybovala právě mezi 30 až 50 procenty. Pro otestování bylo vygenerováno 10 DNA sekvencí pomocí modelu z každé desáté epochy učení od 50. až po 150. epochu a navíc ještě z 200. epochy.

Z grafu na obrázku 3.3 lze vyčíst, že 60. epocha si pro generování DNA sekvencí vedla v porovnání s ostatními epochami mnohem lépe. Dále bylo tedy pracováno s modelem učeným po dobu šedesáti epoch.



Obr. 3.3: Rozdělení podobnosti DNA sekvencí vygenerovaných v různých epochách s DNA sekvencemi z tréninkové množiny

4 Doladění difúzního modelu

K funkci dekodéru, který z vstupní DNA sekvence vygenerované pomocí doladěného GPT-2 modelu vygeneruje obrázek odpovídajícího živočicha, byla vybrána třída neuronových sítí difúzní modely, jejichž fungování je popsáno v sekci 1.3. Difúzní modely jsou v poslední době na vrcholu moderních generativních modelů. Jsou totiž schopny generovat rozmanitá data, konfigurace na učení není náročná a doučení může úspěšně proběhnout i na poměrně malé trénovací množině. Z těchto důvodů se tedy difúzní modely hodí pro tuto práci.

Stejně jako v případě transformátorového modelu je výhodné použít již předučený model a doučit ho na vlastních datech. K tomuto účelu lze použít difúzní model *Stable diffusion*[1] od společností Runway, CompVis a Stability AI.

4.1 Stable diffusion

Model stable diffusion je generativní model, který ze vstupní textové pobídky generuje odpovídající obrázek. Tento model byl naučen na trénovací množině LAION-5B[12], což je 5 miliard párů obrázků a jejich textových popisků stažených z internetu.

Model v základní podobě využívá k zakódování textu do vektorové podoby textový enkodér CLIP[13] od společnosti OpenAI. Tento textový enkodér podporuje příliš krátkou maximální délku vstupní sekvence. Přesně 77 tokenů. To je další z důvodů, proč byl využit GPT-2 model pro doučení na DNA sekvencích. V případě této práce byl tedy místo CLIP enkodéru použit vlastní doučený GPT-2 model.

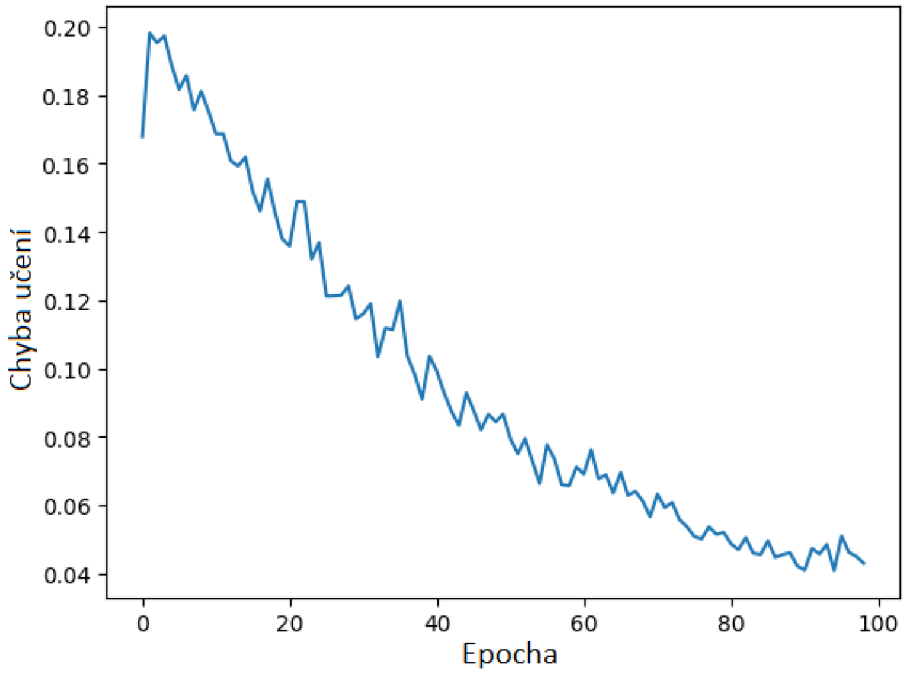
4.1.1 Doučení Stable diffusion

Proces učení probíhá tak, že nejdříve pomocí předučeného VAE obrázek z tréninkové množiny zakódován do latentní reprezentace nižších rozměrů, čímž jsou sníženy výpočetní nároky pro další operace. Dále je navzorkován šum z náhodného kroku zašumění a tento šum je přidán k obrázku z tréninkové množiny (dopředný proces difúze).

Dále je tento zašuměný obrázek, který je převeden do latentního prostoru, spolu s číslem kroku zašumění a zakódovanou DNA sekvencí (zakódování pomocí transformátorového modelu ze sekce 3) předán U-NET difúzního modelu, který odhadne složku šumu (reverzní proces difúze), která byla přidána v dopředném kroku difúze.

Dále jelikož je složka přidaného šumu známa, je vypočítána chyba odhadu šumu. Chyba je vypočítána jako střední kvadratická chyba složky šumu odhadnuté a opravdové. Jelikož jsou obrázky ve formátu RGB, tak je vypočítána střední kvadratická

chyba pro každý pixel každé barevné složky. Dále jsou pro každou barevnou složku chyby pixelů zprůměrovány, a v posledním kroku jsou zprůměrovány i tyto hodnoty. Tímto získáváme chybu v podobě jediného čísla, která je použita v algoritmu zpětné propagace pro upravení vah U-NET sítě.



Obr. 4.1: Závislost chyby učení doučovaného difúzního modelu na epoše učení

Doučení probíhalo ve 100 epochách, kde každá epocha byla otestována vygenerováním několika obrázků. Tyto obrázky byly poté manuálně prohlíženy. Výsledné obrázky zobrazené v sekci 5 jsou z různých epoch. Nelze říct, po které epoše učení difúzní model dosahuje nejlepších výsledků, protože ve všech epochách generuje nesmyslné obrázky se stejnou kadencí.

4.1.2 Generování obrázků pomocí Stable diffusion

Generování obrázku ze vstupní DNA sekvence probíhá tak, že nejdříve je vygenerována originální DNA sekvence, tak jak je popsáno v sekci 3.4.1. Stejně jak je získáno zakódování pro vygenerovanou DNA sekvenci je využit GPT-2 model pro vytvoření zakódování prázdné DNA sekvence. Dále je navzorkován šum, ze kterého bude následně postupnými kroky odšumování v reverzním procesu difúze vytvořen obrázek.

V tuto chvíli tedy dochází v 50 krocích k postupnému odšumování obrázku a to tím způsobem, že je vždy na vstup U-NET sítě přivedena daná latentní reprezentace obrázku v daném kroku odšumování a za pomocí dodatečného vstupu zakódované

DNA sekvence je odhadnut šum přidaný k obrázku v daném kroku. Jak již bylo zmíněno, bylo vypočítáno i zakódování prázdné DNA sekvence. Toto zakódování je v každém kroku používáno také k odhadnutí šumu ve stejném obrázku.

Dále výsledný odhad šumu O jako kombinace dvou různých odhadů šum_{DNA} (pomocí DNA sekvence) a šum_{pr} (pomocí prázdné DNA sekvence) je vypočítán jako:

$$O = \text{š}um_{pr} + G * (\text{š}um_{DNA} - \text{š}um_{pr}) \tag{4.1}$$

G je nastavitelný parametr vlivu šumu odhadovaného pomocí DNA sekvence šum_{DNA} na výsledný odhadnutý šum O. V této práci byl parametr G nastaven na hodnotu 8.

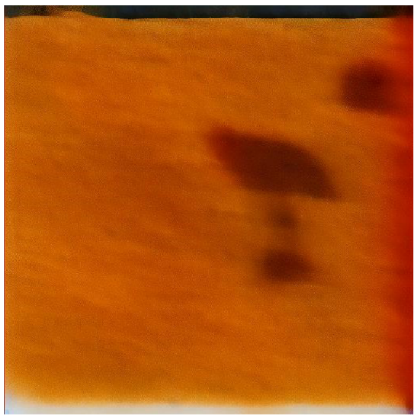
Tento postup generování obrázků pomocí dvou různých odhadů šumu je obvykle používaný. Poprvé byl představen v práci *Classifier-Free Diffusion Guidance*[14]. Tímto postupem odhad šumu nelpí tolik na vstupní DNA sekvenci a díla vygenerovaná pomocí této metody bývají více různorodá.

5 Vygenerované obrázky živočichů

Obr. 5.1: Obrázek vygenerovaný doučeným modelem

Obr. 5.2: Obrázek vygenerovaný doučeným modelem

Na obrázcích 5.1 a 5.2 lze vidět některé z vygenerovaných obrázků. Je nutno říct, že mnoho obrázků vygenerovaných představeným modelem nelze považovat za obrázek živočicha. Takto vygenerovaný obrázek lze vidět na obrázku 5.3.



Obr. 5.3: Obrázek vygenerovaný doučeným modelem

6 Porovnání rekonstrukcí živočichů z testovací množiny

Obr. 6.1: Rekonstrukce rorýse obecného

Obr. 6.2: Fotografie rorýse obecného

Pro otestování funkčnosti modelu byla vytvořena testovací množina živočichů z několika řádů, které se nacházejí stejně jako řád pěvci v třídě ptáci. Jeden z těchto řádů jsou svišťouni. Na obrázku 6.1 lze vidět zástupce svišťounů, rorýse obecného.

DNA sekvence cyklooxygenázy-1 živočichů z testovací množiny byly zakódovány pomocí doučeného GPT-2 modelu a dále předány difúznímu modelu jako vstup.

Na obrázku 6.1 lze vidět rekonstrukci rorýse obecného a na obrázku 6.2 reálnou podobu tohoto živočicha. Tato rekonstrukce je jedna z těch povedenějších. Lze pozorovat vykreslený tvar rekonstruovaného ptáka a postavení živočicha mezi listy stromu. Ovšem často se stávalo, že vygenerované rekonstrukce nešlo s reálnými obrázky porovnávat. Úspěšnost vygenerování použitelné rekonstrukce byla pozorována na asi 15%.

7 Zhodnocení modelu

Při testování doučeného GPT-2 modelu na generování DNA sekvencí bylo zjištěno, že vybraná epocha sice generuje originální obrázky, ale často vygeneruje DNA sekvenci, kterou již dříve vygenerovala. Tomuto by se nejspíše dalo předejít použitím větší tréninkové množiny. Pouze 23 unikátních DNA sekvencí COX-1 genu je na naučení modelu pro generování nových dat málo. DNA sekvence jsou z 30 až 50% podobné, tedy nebyla zachycena tak velká oblast dat.

Vliv na správnost doučení GPT-2 modelu měla s velkou pravděpodovbností i skutečnost, že GPT-2 model je předučený na lidském jazyce. Tento model dobře zachycuje sémantiku a kontext ve větách psaných v lidském jazyce. Sémantika DNA sekvencí není stejná, jako sémantika psaného lidského jazyka. Jakožto doučitelný jazykový model je GPT-2 schopný se učit nová syntaktická a sémantická pravidla, proto za hlavní příčinu občasného generování nesmyslných obrázků velikost trénovací množiny.

V důsledku tohoto doučení nedokázal model správně zachytit korelace mezi jednotlivými úseky DNA sekvence, tím pádem zakódování DNA sekvence dostávané z posledního dekodéru (vysvětleno v sekci 3.3) bylo zkreslené, což se promítlo i na doučení difúzního modelu.

Občasné vygenerování obrázku, který nelze považovat za obrázek živočicha je ale také nutno přikládat realitě, že ne všechny vygenerované DNA sekvence by mohly patřit reálnému živočichovi. Tato obtíž by ale pravděpodobně byla alespoň částečně odstraněna, pokud by model byl učen na větší trénovací množině DNA sekvencí. Ale jak již bylo zmíněno v sekci 2.1, byla snaha o nashromáždění DNA sekvencí ze specifické skupiny živočichů, jako je rod pěvců a ne DNA sekvence například z celé třídy ptáků. Bohužel v internetových databázích nebyla úspěšně nalezena skupina splňující tyto podmínky větší než použitý řád pěvců i přesto, že řád pěvci obsahuje více než polovinu všech zástupců třídy ptáci.

8 Závěr

Představená metoda rekonstruování vymřelých živočišných druhů využívá transformátorový model GPT-2 doučený na vybraných DNA sekvencích ke generování originálních sekvencí. Dále je difúzní model *Stable diffusion* doučen na párech obrázků žijícího živočicha a jejich zakódovaných DNA sekvencí. Metoda generování DNA sekvencí dokáže úspěšně generovat originální DNA sekvence, které se částečně (30-50%) podobají DNA sekvencím z tréninkové množiny. Toto rozložení vzorků podle podobnosti je vyobrazeno na obrázku 3.3. Touto podobností byla zajištěna příbuznost vygenerovaného živočicha s živočichy z tréninkové množiny. Doučený model GPT-2 byl také využit pro zakódování DNA sekvencí, aby mohly být předány difúznímu modelu, který generoval obrázky příslušných živočichů.

Samotné generované obrázky lze považovat za živočicha přibližně v jednom z deseti pokusů generování. Tato úspěšnost je ovlivněna právě experimentálním rázem práce, kdy generování DNA sekvencí je podmíněno pouze doučením GPT-2 modelu na DNA sekvencích žijících živočichů.

Použitím větší tréninkové množiny by pravděpodobně výkon představeného modelu byl mnohem lepší. Při vývoji modelu byla snaha o úzkou podobnost mezi živočichy, jejichž DNA sekvence byly použity pro naučení a to se při testování projevilo jako hlavní problém z důvodu počtu volně dostupných DNA sekvencí v internetových databázích.

Literatura

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. [Online], [cit. 2023-05-20], 2022. arXiv:2112.10752.

[2] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. [Online], [cit. 2023-05-20], 2019. URL: https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[3] Kunfeng Wang, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. Generative adversarial networks:introduction and outlook. *IEEE/CAA journal of automatica sinica*, pages 588–598, 2017. [Online], ročník 4, č.4, [cit. 2022-12-30].

[4] Diederik P Kingma and Max Welling. An introduction to variational auto-encoders. *arXiv.org*, 2019. [Online], Verze 2 (2020), [cit. 2022-12-30]. URL: https://arxiv.org/abs/1906.02691.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. [Online], [cit. 2022-12-30], 2017. URL: https://arxiv.org/abs/1706.03762.

[6] Sarah A. Bates. Base pair. [Online], [cit. 2023-05-20], 2023. URL: https://www.genome.gov/genetics-glossary/Base-Pair#:~:text=%E2%80%8BBase%20Pair&text=The%20two%20strands%20are%20held,and%20cytosine%20pairs%20with%20guanine.

[7] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. [Online], [cit. 2022-12-30], 2022. URL: https://arxiv.org/abs/2204.06125.

[8] Encyclopedia of biological chemistry (second edition). [cit. 2023-05-20]. Academic Press, Waltham, second edition, 2013. URL: https://www.sciencedirect.com/science/article/pii/B9780123786302099904.

[9] Ball Shelley L. Hebert Paul D. N., Cywinska Alina and deWaard Jeremy R. Biological identifications through dna barcodes. [Online], [cit. 2023-05-20], 2003. URL: http://doi.org/10.1098/rspb.2002.2218.

[10] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. [Online], [cit. 2023-05-20], 2021. URL: https://doi.org/10.1093/bioinformatics/btab083.

[11] Luke Salamone. What is temperature in nlp? *lukesalamone.github.io*, 2021. [Online], [cit. 2023-05-20]. URL: https://lukesalamone.github.io/posts/what-is-temperature/#:~:text=Temperature%20in%20NLP%3F-,%F0%9F%90%AD,to%20play%20with%20it%20yourself.

[12] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models. [Online], [cit. 2023-05-20], 2022. arXiv:2210.08402.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. [Online], [cit. 2023-05-20], 2021. arXiv:2103.00020.

[14] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. [Online], [cit. 2023-05-20], 2022. arXiv:2207.12598.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. [Online], Verze 3 (2019), [cit. 2022-12-30]. URL: https://arxiv.org/abs/2006.11239.

[16] Joseph Rocca and Baptiste Rocca. Understanding variational autoencoders (vaes). [Online], [cit. 2022-12-30], 2019. URL: https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73.

[17] Xitong Yang. Understanding the variational lower bound. *xyang35.github.io*, 2017. [Online], [cit. 2022-12-30]. URL: https://xyang35.github.io/2017/04/14/variational-lower-bound/.

[18] Lilian Weng. What are diffusion models? *lilianweng.github.io*, 2021. [Online], [cit. 2022-12-30]. URL: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[19] Ryan O'Connor. Introduction to diffusion models for machine learning. [Online], [cit. 2022-12-30], 2022. URL: https://www.assemblyai.com/blog/diffusion-models-for-machine-learning-introduction.

[20] Ryan O'Connor. Minimagen - build your own imagen text-to-image model. [Online], [cit. 2022-12-30], 2022. URL: https://www.assemblyai.com/blog/minimagen-build-your-own-imagen-text-to-image-model.

[21] Daniel Ibanez. Transformer text embeddings. [Online], [cit. 2022-12-30], 2022. URL: https://www.baeldung.com/cs/transformer-text-embeddings.

[22] Sekvence nukleové kyseliny. [Online], [cit. 2023-05-20], 2022. URL: https://cs.wikipedia.org/wiki/Sekvence_nukleov%C3%A9_kyseliny.

Seznam symbolů a zkratek

KL	Kullback-Leibler
ML	Machine learning
UNS	Umělá neuronová síť
VAE	Variational autoencoder
GAN	Generative Adversarial Network
DNA	Deoxyribonukleová kyselina
API	Application Programming Interface
COX1	Cyklooxygenáza-1

Seznam příloh

Stahovani_COX1.py

Python skript pro stahování COX1 DNA sekvencí z národní lékařské databáze Spojených států amerických (https://www.ncbi.nlm.nih.gov/) na lokálním zařízení.

Stahovani_fotografii_ptaku.py

Python skript pro stahování fotografií pěvců z platformy flickr na lokálním zařízení.

organizmy.txt

Textový soubor organizmů, jejichž fotografie a DNA sekvence je potřeba stáhnout pro vytvoření trénovací množiny.

DNA_tokenizer_BP.ipynb

Google colab notebook zaštiťující převedení DNA sekvencí na tokeny.

Diffusion_uceni_generovani_BP.ipynb

Google colab notebook zaštiťující učení difúzního modelu a následné generování pomocí naučeného modelu.

GPT2_uceni_generovani_BP.ipynb

Google colab notebook zaštiťující učení GPT2 modelu a generování DNA sekvencí.