# Programátorská príručka k modelu variačného autoenkódera

Táto kapitola popisuje implementáciu skriptu `vae_gui.py`, ktorý realizuje trénovanie variačného autoenkódera s vlastným sGUI. Model je navrhnutý na generovanie a rekonštrukciu medicínskych obrazových dát, pričom umožňuje skúmanie rôznych konfigurácií hyperparametrov a generovanie nových vzoriek z latentného priestoru. Implementácia VAE využíva knižnicu PyTorch.

Skript vyžaduje nasledujúce Python knižnice:

- `torch`, `torchvision`
- `matplotlib`, `numpy`, `pandas`
- `PIL (Pillow)`, `opencv-python (cv2)`
- `tkinter`, `csv`, `os`

Potrebné závislosti možno inštalovať príkazom:

```
pip install torch torchvision matplotlib numpy pandas pillow opencv-python
```

## Predspracovanie datasetu

Dataset je pred vstupom do modelu predspracovaný. Po načítaní je na obrázky aplikovaných niekoľko transformácií:

- Nastavenie jednotnej veľkosti obrázkov ($img\_size = 128$), ktoré zabezpečuje, že všetky obrázky vstupujúce do modelu budú mať rovnaké rozmery, čo je dôležité pre správne spracovanie neurónovou sieťou.
- Prevedenie obrázka z formátu sPIL Image (ktorý má formát dĺžka x šírka x kanály) na formát vektora vhodného pre PyTorch (ktorý má tvar kanály x dĺžka x šírka). V prípade RGB sa prevádza obrázok na tvar (3, 128, 128).

- Normalizácia obrázka, ktorá škáluje hodnoty pixelov do rozsahu [-1, 1] odčítaním priemeru ($mean = [0.5]$) a delením štandardnou odchýlkou ($std = [0.5]$).

## Parametre sGUI

sGUI umožňuje konfigurovať nasledujúce hyperparametre:

| **Parameter** | **Typ** | **Predvolená hodnota** | **Popis** |
|---|---|---|---|
| img_size | int | 128 | Veľkosť vstupných obrázkov v pixeloch |
| latent_dim | int | 750 | Veľkosť latentného priestoru |
| batch_size | int | 10 | Veľkosť dávky |
| num_epochs | int | 25 | Počet tréningových epoch |
| $\alpha$ | float | 5.0 | Váha pre rekonštrukčnú stratu |
| $\beta$ | float | 1.0 | Váha pre KL divergenciu |
| $\gamma$ | float | 10.0 | Váha pre stratu rozmazania |
| lr | float | 0.0001 | Rýchlosť učenia |
| in_channels | int | 3 | Počet vstupných kanálov (RGB) |
| depth | int | 4 | Počet konvolučných vrstiev |
| file_path | str | | Cesta k datasetu |
| log_file | str | "./vae_training_log.csv" | Cesta k sCSV logu |
| model_save_dir | str | "./vae_checkpoints" | Cesta pre ukladanie modelov |
| output_dir | str | "./output_images" | Cesta pre ukladanie generovaných obrázkov |

Tabuľka 1: Prehľad nastaviteľných parametrov sGUI

## Implementácia modelu

Model VAE pozostáva z dvoch častí (enkóder a dekóder) a procesu reparametrizácie:

**Enkóder:** Postupne prijíma dávky vstupných snímkov, ktorej veľkosť je určená parametrom *batch_size* a postupne znižuje ich priestorové dimenzie pomocou niekoľkých konvolučných vrstiev (počet určený parameterom *depth*, striedajúcich sa s ReLU funkciami. Veľkosť filtra je nastavená na 4, veľkosť kroku na 2 a výplň na 1. Výstup je vyrovnaný (angl. *flattened*) a spracovaný plne prepojenými vrstvami na výpočet $\mu$ (stredná hodnota) a $\log\sigma^2$ (logaritmus rozptylu latentného priestoru).

**Reparametrizácia**: Po vzorkovaní z latentného priestoru pomocou funkcie *reparameterize* (implementácia triku reparametrizácie, ktorý umožňuje spätnú propagáciu), prechádza latentný vektor ($z$) plne prepojenou vrstvou (*fc_decode*), aby sa opäť preformátoval späť na vhodnú veľkosť pre dekóder.

**Dekóder:** Rekonštruuje pôvodný obrázok z latentnej reprezentácie cez transponované konvolučné vrstvy striedajúce sa s ReLU vrstvami. Veľkosť filtra je nastavená na 4, veľkosť kroku na 2 a výplň na 1. Výstup je normalizovaný cez aktivačnú funkciu $Tanh$ do rozsahu [-1, 1].

Celkovým výstupom siete sú 3 prvky, a to dva vektory - stredná hodnota ($mu$) a logaritmus rozptylu ($logvar$), parametre normálneho rozdelenia pre latentné premenné v rámci VAE - a rekonštruovaný obrázok s rozmermi (*in_channels*, *img_size*, *img_size*).

## Funkcia straty

Celková strata kombinuje tri zložky:

1. **Rekonštrukčná strata:** MSE chyba medzi originálnym a rekonštruovaným obrázkom, vážená parametrom $\alpha$.

2. **KL divergencia:** Divergencia medzi latentným rozdelením a štandardnou normálnou distribúciou, vážená parametrom $\beta$.

3. **Strata rozmazania:** Penalizácia nízkeho rozptylu Laplacianovej variancie, vážená parametrom $\gamma$.

## Priebeh tréningu

Tréning je riadený funkciou *run_vae_training* a obsahuje nasledujúce kroky:

1. Detekcia zariadenia (GPU).
2. Načítanie datasetu cez vlastnú triedu *MedicalImageDataset*.
3. Inicializácia modelu, optimalizátora Adam a škálovača gradientov.
4. Obnovenie stavu z najnovšieho kontrolného bodu (ak existuje).
5. Tréningová slučka na určitý počet epoch *num_epochs*:
   - Výpočet straty.
   - Aktualizácia parametrov.
   - Logovanie stratových metrík po každej epoche do sCSV súboru (číslo epochy a hodnoty).
   - Uloženie kontrolného bodu modelu.
6. Vizualizácia priebehu straty.
7. Generovanie nových dát vzorkovaním náhodných latentných vektorov $z$, a spracuje ich cez dekodér, aby získal rekonštruované obrázky.
8. Vizualizácia generovaných obrázkov v mriežke 8x8 a ich uloženie do priečinku *./output_images*.

Počas tréningu je aplikovaný nástroj z knižnice PyTorch, škálovač gradientu, ktorý zrýchľuje tréningový proces, šetrí pamäťové a časové zdroje a zároveň udržiava presnosť modelu. Dynamicky upravuje rozsah hodnôt spätného šírenia gradientov počas tréningu modelu, aby sa predišlo podtečeniu malých čísel v prípade použitia 16-bitovej presnosti. Pri použití zmiešanej presnosti (*mixed precision*) s `float16` namiesto štandardnej `float32` sa výrazne znižuje pamäťová náročnosť a zvyšuje sa rýchlosť výpočtov, čo je výhodné najmä pri tréningu na grafických kartách. Škálovač automaticky zisťuje vhodnú mierku a v prípade, že dôjde k numerickej nestabilite (napríklad výskyt `NaN`), túto mierku upraví.

## Poznámky

- Uistite sa, že *file_path* ukazuje na adresár s sJPEG obrázkami.
- Parameter *img_size* musí byť deliteľný 2d̂epth, aby sa predišlo nezhodám v rozmeroch VAE.
- Súbor *log_file* by mal mať príponu sCSV. Ak neexistuje, bude vytvorený.
- Ak neexistujú adresáre *model_save_dir* a *output_dir*, tak sú taktiež vytvorené.

Pre ďalšiu pomoc skontrolujte výstup konzoly pre chybové hlásenia alebo konzultujte dokumentáciu PyTorch a Tkinter.