M2 SETI : C4 Fusion

Enseignants : S. Le Hégarat, E. Aldea, R. Reynaud.

TP : techniques de fusion par apprentissage supervisé

Cette séance de TP a pour objectif la mise en œuvre de méthodes de fusion de données multimodales basée sur un apprentissage supervisé.

Pour chaque TP, vous aurez une semaine pour rendre un rapport écrit, où vous justifierez votre implémentation des algorithmes, analyserez l'influence de leurs paramètres, et commenterez les résultats obtenus. Ces commentaires porteront notamment sur la qualité des solutions implémentées et leur comparaison.

1 Environnement

Les codes seront développés en Pytorch à partir du squelette fourni, qui contient des "trous" qui sont les parties à compléter.

2 Données

2.1 Objectif

Le but du TP consiste à réaliser plusieurs types de fusions entre deux modalités différentes, image et audio, pour un jeu de données relativement simple afin de permettre un entraînement de durée raisonnable.

2.2 Dataset AVMNIST

Les données sont à télécharger au lien suivant :

`https://drive.google.com/file/d/1KvKynJJca5tDtI5Mmp6CoRh9pQywH8Xp/view?usp=sharing`

Il s'agit des chiffres, la différence par rapport à MNIST étant qu'il y a également une information audio concernant la classe, en plus de l'information image.

Utilisez le dataloader fourni et regardez d'abord la structure des données. Il s'agit d'images en resolution 28×28 et des signaux audio sous la forme des spectrogrammes de résolution 112×112

Affichez un tableau d'images et leurs annotations. Qu'en pensez vous en terme de difficulté par rapport à MNIST ?

2.3 Performances des classifieurs unimodaux

On commence par implémenter et entraîner des architectures simples pour classifier à partir des informations unimodales.

Définissez un modèle LeNet5 et entraînez le sur les données image. La répartition des données est 55k entraînement, 5k validation, 10k test. Notez le nombre de paramètres et l'accuracy obtenue.

De même, proposez et entraînez un modèle inspiré de LeNet5 sur les données audio. Notez le nombre de paramètres et l'accuracy obtenue.

2.4 Fusion en amont

Proposez et entraînez un modèle qui récupère en entrée les deux types de données brutes. Évitez l'utilisation des couches FC tout de suite en entrée du réseau car cela demanderait beaucoup de paramètres à optimiser. Normalement, l'accuracy obtenue devrait être supérieure à celles obtenues aux questions précédentes. Quel est le nombre de paramètres du nouveau modèle ?

2.5 Fusion intermédiaire

A partir des deux architectures séparées de type LeNet proposées initialement, concaténez les vecteurs latents flattened obtenus après les convolutions, afin d'obtenir un seul modèle à entraîner conjointement, sans fusioner les données brutes en amont. Quel est le nombre de paramètres et la performance de ce modèle ?

2.6 Comparaison

Comparez la performance obtenue avec celle du code suivant, est-ce que les résultats sont cohérents si on regarde l'accuracy et la taille du modèle ?

```
channels = 6
encoders = [LeNet(1, channels, 3).cuda(), LeNet(1, channels, 5).cuda()]
head = MLP(channels*40, 100, 10).cuda()

fusion = Concat().cuda()

train(encoders, fusion, head, traindata, validdata, 30,
      optimtype=torch.optim.SGD, lr=0.1, weight_decay=0.0001)

print("Testing:")
model = torch.load('best.pt').cuda()
test(model, testdata, no_robust=True)
```