Rapport de Projet : Classification des Images de Chiens avec TensorFlow

1. Contexte du Projet

Ce projet aborde un problème de classification multi-classes : identifier la race de chiens à partir d'images. Ce problème repose sur l'analyse de données provenant du concours Kaggle Dog Breed Identification, qui contient plus de 10 000 images étiquetées pour 120 races distinctes.

2. Problème

- **Type de tâche** : Classification multi-classes.
- **Données d'entrée** : Images de chiens.
- **Données de sortie** : Une prédiction sur les 120 races possibles.
- **Objectif** : Développer un modèle performant capable de prédire correctement la race d'un chien avec une précision élevée.

3. Méthodologie

3.1 Préparation des Données

1. **Chargement des données** :
    - Les images et leurs étiquettes sont extraites des fichiers Kaggle.
    - Les chemins des fichiers et labels correspondants sont organisés pour un accès facile.
2. **Prétraitement des images** :
    - **Redimensionnement** : Toutes les images sont redimensionnées (224x224).
    - **Normalisation** : Les pixels sont convertis en valeurs flottantes comprises entre 0 et 1.
    - **Augmentation de données** :
        - Rotation aléatoire.
        - Flips horizontaux.

        - Ajustement de la luminosité/contraste.
3. **Création des ensembles de données** :
    - Séparation en trois ensembles :
        - **Entraînement (80%)** : Utilisé pour ajuster les paramètres du modèle.
        - **Validation (20%)** : Permet de surveiller les performances pendant l'entraînement.
        - **Test (100%)** : Utilisé pour évaluer les performances finales.
4. **Batching des données** :
    - Les ensembles sont divisés en lots grâce à TensorFlow :

```
batch_size = 32

train_dataset = (tf.data.Dataset.from_tensor_slices((train_images, train_labels))

                 .map(preprocess_image)

                 .shuffle(1000)

                 .batch(batch_size)

                 .prefetch(tf.data.AUTOTUNE))
```

3.2 Modélisation

1. **Approche choisie : Apprentissage par transfert**
    - Utilisation de modèles préentraînés disponibles sur TensorFlow Hub.
    - Exemple : `MobileNetV2`, qui est performant en termes de précision et d'efficacité.
2. **Structure du modèle** :
    - Une couche principale basée sur un modèle préentraîné.
    - Une couche dense finale avec 120 neurones (une pour chaque classe), activation `softmax`.
    - Exemple de code :

```
model = tf.keras.Sequential([
```

```
    hub.KerasLayer("https://tfhub.dev/google/imagenet/efficientnet_v2_imagenet1k_b0/feature_vector", trainable=True),
    tf.keras.layers.Dense(120, activation='softmax')
])
```

Compilation :

- **Optimiseur** : `Adam` avec un taux d'apprentissage initial de 0.001.
- **Fonction de perte** : `categorical_crossentropy` pour la classification multi-classes.
- **Métriques** : `accuracy` pour évaluer la performance.

Entraînement :

- Réalisé sur un nombre d'époques prédéfini (par exemple, 20).
- Utilisation de l'**EarlyStopping** pour arrêter l'entraînement si la performance ne s'améliore pas après 3 itérations :

```
early_stopping = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss', patience=3, restore_best_weights=True
)

history = model.fit(train_dataset, validation_data=val_dataset, epochs=20, callbacks=[early_stopping])
```

3.3 Évaluation

1. **Sur l'ensemble de test** :
    - Calcul de la précision globale.
    - Analyse qualitative : vérification des images mal classées.
2. **Métriques supplémentaires** :
    - Matrice de confusion.

4. Résultats

- **Performance attendue** : Précision supérieure à 99% sur l'ensemble de test.
- **Modèle sauvegardé** :
    - Format : `SavedModel` ou `.h5`.
    - Exemple de sauvegarde :

6. Conclusion

Ce projet démontre comment les modèles préentraînés peuvent être utilisés efficacement pour des tâches de classification complexes. Avec une approche méthodique (prétraitement, modélisation, ajustement), il est possible d'obtenir un modèle performant sans nécessiter une grande puissance de calcul.

Si vous avez des points spécifiques que vous souhaitez approfondir ou modifier dans ce rapport, n'hésitez pas à me le demander !