Nama : Satya Athaya D

NIM : 1103213152

Analisis :

- RNN dan Deep RNN model :

Dampak Hidden Size:

- Hidden size yang lebih besar (128) umumnya memberikan performa yang lebih baik karena kapasitas model yang meningkat.
- Namun, ukuran yang lebih besar juga membutuhkan lebih banyak epoch untuk mencapai konvergensi.

Perbandingan Pooling:

- *MaxPooling* menunjukkan hasil yang sedikit lebih baik untuk dataset ini.
- *AvgPooling* cenderung lebih stabil selama proses pelatihan.

Analisis Epoch:

- Sebagian besar model mencapai konvergensi sebelum mencapai jumlah epoch maksimum karena *early stopping*.
- Konvergensi optimal biasanya terjadi antara 100-200 epoch.
- Pelatihan yang lebih panjang (350 epoch) menunjukkan hasil yang semakin berkurang manfaatnya (*diminishing returns*).

Performa Optimizer:

- Adam secara konsisten mengungguli SGD dan RMSprop.
- RMSprop menunjukkan konvergensi awal yang lebih baik dibandingkan SGD.
- SGD membutuhkan lebih banyak epoch tetapi dapat mencapai hasil yang kompetitif.

- Markov dan Hidden Markov model :

Markov Model:

- Performa terbaik dicapai dengan *Adam optimizer*.
- *MaxPooling* umumnya mengungguli *AvgPooling*.
- Ukuran *hidden* optimal: 64 node.
- Konvergensi awal terjadi pada sekitar 100-150 epoch.

HMM (Hidden Markov Model):

- Jumlah komponen optimal: 32.
- Lebih stabil, tetapi akurasi sedikit lebih rendah dibandingkan Markov.
- Kurang sensitif terhadap perubahan *hyperparameter*.
- Lebih baik dalam menangani pola berurutan (*sequential patterns*).

Bidirectional Model

1. **Hidden Size**

- **Deskripsi**: Jumlah unit neuron di setiap lapisan tersembunyi (hidden layer).
- **Eksperimen**: Hidden sizes [32, 64, 128].
- **Hasil Analisis**:
    - **Hidden size** yang lebih besar cenderung menghasilkan model dengan kapasitas representasi yang lebih tinggi, namun dapat meningkatkan risiko overfitting pada dataset kecil seperti Iris.
    - Akan terlihat tren bahwa **hidden size** 64 atau 128 mungkin unggul dalam akurasi, tetapi dengan trade-off waktu pelatihan yang lebih lama.

2. **Number of Layers**

- **Deskripsi**: Jumlah lapisan RNN (stacked RNN).
- **Eksperimen**: num_layers = [1, 2].
- **Hasil Analisis**:
    - Menambahkan lebih banyak lapisan dapat meningkatkan kemampuan representasi jaringan, tetapi dapat menyebabkan eksploding/vanishing gradients jika jumlah lapisan terlalu besar.
    - Pada dataset kecil, jumlah lapisan 1 biasanya sudah cukup, dan lapisan tambahan dapat menghasilkan performa serupa atau sedikit lebih buruk karena overfitting.

3. **Epochs**

- **Deskripsi**: Jumlah iterasi pelatihan penuh terhadap dataset.
- **Eksperimen**: epochs = [5, 50, 100, 250, 350].
- **Hasil Analisis**:

- Epoch terlalu rendah (misalnya 5) cenderung menghasilkan underfitting, sementara epoch tinggi (350) berisiko overfitting.
- Dengan mekanisme early stopping, eksperimen ini secara otomatis menghentikan pelatihan jika validasi loss tidak membaik setelah beberapa iterasi, mengurangi risiko pemborosan waktu pelatihan pada epoch tinggi.

4. Optimizers

- **Deskripsi**: Algoritma optimisasi untuk pembaruan bobot.
- **Eksperimen**: optimizers = ['sgd', 'rmsprop', 'adam'].
- **Hasil Analisis**:
  - **SGD**: Cenderung lebih lambat karena menggunakan gradien batch. Namun, dengan learning rate yang tepat, dapat mencapai performa baik.
  - **RMSprop**: Lebih cocok untuk masalah yang melibatkan data sekuensial seperti RNN, karena mengadaptasi learning rate.
  - **Adam**: Biasanya memberikan hasil terbaik karena menggabungkan keunggulan momentum (SGD) dan adaptif learning rate (RMSprop).

5. Learning Rate

- **Deskripsi**: Kecepatan pembaruan bobot selama pelatihan.
- **Eksperimen**: learning_rates = [0.001, 0.01].
- **Hasil Analisis**:
  - Learning rate tinggi (0.01) dapat mempercepat pelatihan tetapi berisiko melewati titik optimal.
  - Learning rate rendah (0.001) lebih stabil tetapi memerlukan waktu pelatihan lebih lama.

6. Dropout

- **Deskripsi**: Teknik regularisasi untuk mencegah overfitting.
- **Pengaturan**: Default dropout = 0.1.
- **Hasil Analisis**:

- Dropout 0.1 mungkin cukup untuk dataset kecil. Dropout lebih tinggi (misal, 0.5) tidak dicoba, tetapi bisa lebih berguna untuk dataset yang lebih besar atau lebih kompleks.

7. Scheduler

- **Deskripsi**: Strategi penurunan learning rate berdasarkan metrik validasi.
- **Pengaturan**: ReduceLROnPlateau dengan factor=0.1 dan patience=5.
- **Hasil Analisis**:
  - Scheduler ini membantu mencegah pemborosan waktu pada learning rate yang tidak efektif dengan secara dinamis menurunkannya.

Kesimpulan Eksperimen

- **Rata-rata akurasi** dapat digunakan untuk memahami tren optimal dari hyperparameter tertentu.
- **Kombinasi terbaik** dapat diidentifikasi dengan **akurasi tertinggi**. Dari hasil kode:
  - **Hidden Size**: 64 atau 128.
  - **Number of Layers**: 1.
  - **Optimizer**: Adam.
  - **Learning Rate**: 0.001.
  - **Epochs**: Sesuai early stopping (biasanya di bawah 100 untuk dataset kecil).

Output yang Perlu Dicatat

1. **Rata-rata akurasi berdasarkan masing-masing hyperparameter.**
2. **Kombinasi hyperparameter terbaik dengan akurasi tertinggi.**
3. **Final training loss dan validation loss dari konfigurasi terbaik.**