Nama : Eko Putra Nugraha

NIM : 1103213212

RNN dan Deep RNN model

Eksperimen yang dijalankan menghasilkan serangkaian model RNN dan Deep RNN dengan berbagai konfigurasi hyperparameter, termasuk ukuran hidden layer (32, 64, dan 128), jenis pooling (max dan avg untuk Deep RNN), jumlah epoch training (5, 50, 100, 250, 350), serta optimizer yang digunakan (SGD, RMSProp, dan Adam). Dari visualisasi yang dihasilkan, terlihat bahwa variasi hyperparameter memiliki dampak signifikan terhadap performa model. Model dengan ukuran hidden layer yang lebih besar cenderung menghasilkan performa yang lebih baik, tetapi dengan peningkatan waktu training. Selain itu, penggunaan pooling pada Deep RNN juga mempengaruhi hasil, di mana beberapa konfigurasi menunjukkan akurasi yang lebih baik daripada yang lain. Perbedaan optimizer juga menghasilkan performa yang berbeda, beberapa menunjukkan konvergensi lebih cepat dan akurasi yang lebih tinggi dibandingkan yang lain.

Jumlah epoch training juga berperan penting dalam performa model. Grafik loss dan akurasi menunjukkan bahwa pada jumlah epoch yang lebih kecil, model belum mencapai konvergensi optimal. Sebaliknya, model dengan jumlah epoch yang lebih besar cenderung menunjukkan peningkatan performa, meskipun terkadang ada risiko overfitting. Callback early stopping yang diimplementasikan membantu mencegah overfitting dan menghentikan proses training ketika tidak ada peningkatan signifikan pada test loss setelah beberapa epoch, sehingga membantu menghemat waktu dan sumber daya komputasi. Learning rate scheduler juga berperan dalam mengoptimalkan training dengan menyesuaikan learning rate saat plateau loss tercapai.

Visualisasi perbandingan akurasi test terbaik dari semua konfigurasi model memberikan gambaran yang jelas tentang konfigurasi mana yang menghasilkan performa paling baik. Bar chart tersebut memungkinkan kita untuk dengan mudah mengidentifikasi kombinasi hyperparameter yang menghasilkan akurasi tertinggi pada data testing. Dari hasil perbandingan tersebut, dapat disimpulkan bahwa kombinasi tertentu (misalnya, ukuran hidden layer yang lebih besar, optimizer yang spesifik, atau jumlah epoch yang optimal) cenderung lebih unggul daripada kombinasi lain dalam tugas klasifikasi income ini. Analisis ini membantu kita untuk memilih konfigurasi model yang paling sesuai untuk permasalahan yang dihadapi dan mengoptimalkan performa model untuk tugas klasifikasi income.

Markov model dan Hidden Markov Model

Pengaruh Hidden Size dan Pooling: Eksperimen dengan berbagai ukuran hidden size (32, 64, dan 128) dalam lapisan RNN menunjukkan pengaruh signifikan terhadap kemampuan model untuk mempelajari representasi data. Hidden size yang lebih besar, seperti 128, cenderung memberikan fleksibilitas model yang lebih tinggi dan berpotensi menangkap pola yang lebih kompleks, meskipun ini juga meningkatkan risiko overfitting dan kebutuhan komputasi. Perbandingan antara *max pooling* dan *average pooling* mengungkapkan bahwa *max pooling* cenderung memberikan performa yang sedikit lebih baik, kemungkinan karena kemampuannya untuk menyoroti fitur yang paling relevan dalam output RNN. Namun, perbedaan ini mungkin tidak signifikan dalam semua konfigurasi dan perlu dievaluasi lebih lanjut berdasarkan dataset dan tugas tertentu.

Dampak Jumlah Epoch dan Optimizer: Variasi jumlah epoch (5, 50, 100, 250, dan 350) menunjukkan bagaimana iterasi pelatihan memengaruhi kinerja model. Epoch yang lebih sedikit cenderung

membuat model underfit dengan hasil yang kurang optimal, sedangkan epoch yang terlalu banyak dapat menyebabkan overfitting yang membuat model kurang baik saat mengeneralisasi data yang tidak terlihat. Penggunaan *early stopping* dengan kesabaran 20 sangat membantu untuk mencegah overfitting dan memastikan bahwa training berhenti ketika test loss tidak lagi membaik. Perbandingan antara optimizer *SGD*, *RMSprop*, dan *Adam* menunjukkan bahwa *Adam* cenderung mencapai hasil yang lebih baik dan konvergen lebih cepat, ini karena metode *adaptive learning rate* yang ditawarkan. *SGD* dengan *learning rate scheduler* juga menghasilkan hasil yang baik, namun dengan laju konvergensi yang lebih lambat.

Interaksi Antar Parameter: Dari hasil eksperimen, terlihat bahwa performa terbaik model tidak hanya bergantung pada satu parameter saja, namun dari interaksi yang sinergis antara berbagai parameter. Contohnya, model dengan *hidden size* besar mungkin memerlukan epoch yang lebih sedikit jika dipadukan dengan optimizer *Adam*, sedangkan model dengan *hidden size* kecil mungkin memerlukan lebih banyak epoch dan learning rate yang disesuaikan dengan optimizer *SGD*. Pola yang diamati menunjukkan bahwa pemilihan parameter yang tepat sangat bergantung pada dataset yang digunakan dan perlu disesuaikan berdasarkan kebutuhan spesifik. Oleh karena itu, eksplorasi yang sistematis dari berbagai konfigurasi, seperti yang dilakukan dalam eksperimen ini, penting untuk mencapai hasil yang optimal.

Bidirectional RNN Model

1. Hidden Size pada RNN: Hidden size dalam RNN menentukan kapasitas representasi model. Ukuran hidden state yang lebih besar memungkinkan model untuk menyimpan lebih banyak informasi tentang urutan input, yang berpotensi menghasilkan kinerja yang lebih baik pada tugas-tugas kompleks. Namun, hidden size yang terlalu besar juga dapat menyebabkan overfitting dan membutuhkan lebih banyak sumber daya komputasi. Sebaliknya, hidden size yang terlalu kecil dapat membatasi kemampuan model untuk mempelajari pola yang kompleks, mengakibatkan underfitting. Oleh karena itu, memilih hidden size yang optimal membutuhkan eksperimen dan penyesuaian untuk menemukan titik tengah yang baik antara kapasitas model dan kemampuan generalisasi. Pada kode yang kita buat, kita membandingkan beberapa hidden size, yaitu 32, 64, dan 128, untuk melihat bagaimana dampaknya terhadap kinerja model pada dataset Iris.

2. Pooling (MaxPooling dan AvgPooling): Pooling merupakan teknik yang digunakan untuk mengurangi dimensi representasi fitur dan menangkap informasi yang paling relevan. Dalam konteks RNN, pooling diterapkan setelah lapisan RNN untuk merangkum informasi dari seluruh sequence. MaxPooling mengambil nilai maksimum dari seluruh hidden state pada setiap posisi sequence, yang berguna untuk menangkap fitur yang paling aktif. Sementara itu, AvgPooling mengambil rata-rata dari seluruh hidden state, yang lebih baik dalam menangkap informasi keseluruhan dari sequence. Pemilihan antara MaxPooling dan AvgPooling bergantung pada karakteristik data dan tugas yang dihadapi. Dalam konteks dataset Iris, kita melihat bagaimana penggunaan MaxPooling dan AvgPooling mempengaruhi kinerja model karena pada dataset Iris tidak terdapat sequence data seperti pada NLP atau timeseries, sehingga penggunaan sequence-wise pooling bisa dibilang kurang tepat.

3. Epoch, Optimizer, dan Learning Rate Scheduler: Epoch adalah jumlah iterasi lengkap melalui seluruh dataset training. Jumlah epoch yang lebih banyak dapat membantu model untuk mempelajari pola yang lebih baik, tetapi juga dapat meningkatkan risiko overfitting. Kita melakukan perbandingan dengan epoch 5, 50, 100, 250, dan 350 untuk melihat bagaimana pengaruh jumlah epoch terhadap training. Selain itu, optimizer berperan dalam mengarahkan proses pembelajaran dengan memperbarui bobot model berdasarkan gradien loss function. Dalam kode yang ada, kita

membandingkan SGD, RMSProp, dan Adam. SGD cenderung lambat dan rentan terhadap oscilasi, sementara RMSProp dan Adam memiliki mekanisme learning rate adaptif yang memungkinkan konvergensi yang lebih cepat. Terakhir, learning rate scheduler seperti ReduceLROnPlateau membantu menyesuaikan learning rate selama training berdasarkan kinerja validasi, yang dapat meningkatkan efisiensi dan stabilitas proses pembelajaran dan membantu menghindari overfitting. Penggunaan early stopping juga menjadi langkah penting dalam mencegah overfitting ketika epoch terlalu banyak. Dengan membandingkan parameter-parameter ini secara sistematis, kita dapat menemukan kombinasi terbaik yang menghasilkan kinerja model optimal pada dataset Iris.