I. Latar Belakang

Image size diubah menjadi 224 x 224 karena ukuran tersebut direkomendasikan saat ingin menggunakan pretrain model. Sebelum membuat model ini ada banyak percobaan yang dilakukan terkait transfer learning ini. Walaupun hal ini dapat memberatkan komputasi saat training, tetapi size inilah yang paling kompatibel untuk menggunakan pretrain model.

Dari yang diajarkan, saat ingin menggunakan pretrain model, disarankan untuk melakukan preprocess input dan freeze layer untuk mengurangi beban komputasi dan menyesuaikan pretrain model dengan classifier data yang ingin digunakan. Namun, pada kasus ini setelah saya melakukan freeze layer dan preprocess input atau salah satunya, hasil f1 dan akurasi validasi masih jauh dibandingkan dengan jika layer pretrain model tidak di-freeze ataupun dilakukan preprocess input. Setelah saya membaca beberapa artikel, ternyata konsekuensi dari tidak melakukan freeze layer pada pretrain model adalah daya komputasi yang sangat besar sehingga training dataset menjadi jauh lebih lama, dan hal ini memang terjadi. Selain itu, dengan merubah size image yang awalnya 28 x 28 menjadi 224 x 224 juga menambah daya komputasi saat training. Alhasil training sangat membutuhkan waktu yang lama. Namun, karena lomba lebih berfokus pada f1 dan akurasi dari model dan tidak memedulikan cepat atau lambatnya model dalam melakukan training, jadi saya lebih memilih model dengan validasi f1 dan akurasi yang lebih tinggi. Akhirnya, saya mencoba menggunakan pretrain model tanpa melakukan freezing layer dan tanpa melakukan preprocess input. Hasil yang saya peroleh cukup mengejutkan, akurasi validasi dan f1 validasinya menjadi jauh lebih tinggi, hingga mencapai angka 93-94%. Nilai f1 dan akurasi yang tidak jauh berbeda berarti datanya juga tidak bias sehingga model dapat mengenali semua classnya dengan baik. Dari sumber (https://datascience.stackexchange.com/questions/30659/what-are-the-consequences-of-not-freezing-layers-in-transfer-learning) yang saya baca juga saya mendapat wawasan baru dimana jika akurasi menjadi lebih tinggi daripada saat melakukan freeze dan preprocess input, maka input image yang ada (Body Organs CT Scan) jauh berbeda dengan yang ada di imagenet, sehingga jika melakukan freeze, maka akan menurunkan akurasi model. Tidak melakukan freeze layer maka semua layer yang ada di pretrain model akan digunakan untuk training, hal utama yang menjadi perhatian saat tidak melakukan freeze layer adalah terjadi overfitting karena model sangat kompleks. Namun, dalam hal ini overfit tidak terjadi, melainkan model bisa mendapatkan akurasi validasi dan f1 validasi yang maksimal. Oleh alasan – alasan inilah saya tidak melakukan freeze layer dan preprocess input pada pretrain model.

Berikutnya, saya juga telah melakukan trial and error untuk masalah augmentasi data. Dari hasil yang saya peroleh, saat saya menambahkan data augmentation pada image generator dengan melakukan horizontal flip atau vertical flip atau zoom range atau ketiganya, akurasi model juga menjadi turun dibandingkan dengan tidak melakukan data augmentation. Selain itu, data augmentasi adalah salah satu teknik untuk mencegah terjadinya overfitting, tetapi saya juga sudah menambahkan dropout pada arsitektur model yang mencegah terjadinya overfitting juga, sehingga selama model tidak overfitting dan mampu memberikan akurasi dan f1 yang baik maka data augmentation

tidak diperlukan. Dari sini saya bisa simpulkan jika banyak gambar pada dataset dan kompleksitas model yang saya bangun sudah sesuai sehingga mampu bekerja dengan sangat baik sehingga model bisa menemukan pola yang ada pada gambar dan dapat menghasilkan hasil training-validation accuracy dan f1 yang tinggi.

Saya juga menggunakan teknik hyperparameter tuning secara manual karena keterbatasan spesifikasi laptop saya. Saya melakukan trial and error pada parameter saat model di-compile, tepatnya pada learning rate dari 0.01 hingga 0.00001 dan hasil akhir menunjukkan jika nilai 0.0001 yang terbaik. Selain itu, saya juga menggunakan categorical crossentropy karena class terdapat 11 class dalam dataset (> 1 class).

Terdapat 5 pretrain model yang saya coba, yaitu VGG16, MobileNet, MobileNetV2, Resnet50, dan EfficientNet. Dari pengalaman saya, 5 model inilah yang dapat memberikan hasil yang maksimal dibanding model pretrain lainnya. Dari 5 pretrain model ini, Saya menggunakan 3 pretrain model yang memberi validasi akurasi dan f1 tertinggi yaitu VGG16 (94,6%), MobileNet (94,4%), dan ResNet50 (94,6%). Hyperparameter tuning manual yang saya pakai disini adalah untuk menentukan apakah fully connected layer lebih cocok menggunakan GlobalAveragePooling2D atau Flatten, dan hasilnya lebih baik menggunakan Flatten. Saya juga mencoba activation function menggunakan sigmoid, dan ternyata softmax mampu memberikan hasil yang lebih baik. Selain itu, saya perlu mempertimbangkan nilai neuron pada Dense layer dan menemukan nilai 1024 (wide network) lebih baik daripada membuat 2-3 Dense layer yang berbeda dengan nilai neuron yang lebih kecil dari 1024 (deep network). Saat melakukan training, saya juga menggunakan callback untuk menyimpan weight dari model (.h5) untuk model dengan akurasi validasi terbaik saja. Kemudian saya melakukan load weight untuk 3 model tersebut. Terakhir, saya juga melakukan ensemble learning pada 3 model tersebut untuk mendapatkan akurasi dan f1 yang lebih tinggi dari 94%. Alasan saya memilih 3 model adalah agar tidak terjadi prediksi bias akibat hasil dari 2 model saja. Ibarat juri harus berjumlah ganjil, hal ini juga saya implementasikan dalam ensemble learning ini. Dari sini saya bisa memperoleh f1 score test terbaik dari semua model yang pernah saya buat.

II. Metode

Karena saya menggunakan 3 pretrain model yang berbeda, maka saya akan memulai langsung ke bagian classification head yang saya buat di akhir arsitektur, tepatnya setelah pretrain model. Untuk ketiga model, saya menggunakan classification head yang sama, yakni dengan menggunakan Flatten untuk mengubah feature map menjadi vektor. Kemudian menambahkan fully connected layer Dense dengan neuron sebanyak 1024 dan menambahkan dropout 0.5 untuk mencegah terjadinya overfitting. Terakhir, saya menambahkan output layer dengan softmax activation function untuk memprediksi 11 kelas pada dataset. Berikut ini penjelasan untuk setiap metode atau arsitektur pada setiap pretrain model yang saya gunakan:

i) VGG16

Gambar yang menjadi input memiliki dimensi (224, 224, 3). Dua layer pertama memiliki channel sebanyak 64 dari 3 x 3 ukuran filter dengan padding same.

Kemudian, terdapat max pool later dengan stride (2, 2), dua laayer memiliki convolution layer dengan 128 ukuran filter dan (3, 3). Hal ini diikuti dengan max pool layer lagi dengan stride (2, 2), sama seperti sebelumnya. Kemudian terdapat 2 convolution layer dengan ukuran filter (3, 3) dan terdapat 256 filter di dalamnya. Setelah itu, terdapat 2 set 3 convolution layer dan 1 max pool layer. Masing – masing memiliki 512 filter berukuran (3, 3) dengan padding same. Kemudian gambar diteruskan ke stack kedua dari convolution layers. Dalam layer ini, convolution dan max pool layer menggunakan filter 3 x 3, bukan 11 x 11 seperti AlexNet maupun 7 x 7 seperti ZF-Net. Beberapa layer menggunakan piksel 1 x 1 untuk memanipulasi jumlah input channels. Sementara itu, padding same digunakan setelah tiap convolution layer dijalankan untuk mencegah fitur spatial dari suatu gambar.

Kemudian model akan mendapatkan output sebesar (7, 7, 512). Output akan di flatten untuk membuat feature vector (1, 25088). Terdapat 3 fully connected layer, yang pertama akan mengambil input dari feature vector terakhir dan memberikan output vector sebesar (1, 4096), begitu juga dengan layer kedua. Namun, layer ketiga akan memberikan output 1000 channel untuk mengklasifikasikan 1000 class. Semua hidden layer menggunakan ReLU sebagai activation functionnya karena memiliki komputasi yang lebih efisien.

ii) MobileNet

MobileNet adalah model yang melakukan filter gambar sama seperti CNN, tetapi dengan pendekatan yang berbeda. MobileNet dibangun di atas depthwise seperable convolution layer dan memiliki ukuran input gambar sebesar 224 x 224 x 3, sama seperti kebanyakan pretrain model. Setiap depthwise seperable convolution layer mempunyai depthwise convolution dan pointwise convolution, hal inilah yang membedakan MobileNet dengan arsitektur CNN pada umumnya. Hal ini juga dapat meningkatkan efisiensi CNN untuk memprediksi gambar dan karenanya MobileNet dapat digunakan untuk perangkat mobile pula. Karena MobileNet banyak mengurangi waktu perbandingan dan pengenalan gambar, maka MobileNet dapat memberikan respons yang lebih baik dalam jangka waktu yang pendek.

Daripada itu, MobileNet memiliki 28 layer yang dibagi atas depthwise dan pointwise convolution layer itu sendiri. Normalnya, MobileNet mempunyai 4.2 juta parameter yang dapat dikurangi lebih banyak dengan mengatur hyperparameternya secara tepat.

iii) ResNet50

ResNet50 mengalami perubahan kecil yaitu menambahkan 1 x 1 convolution layer. Elemen awal pada ResNet50 adalah convolution layer dengan ukuran kernel sebesar 7 * 7 dan 65 kernel yang berbeda semua dengan stride berukuran 2. Kemudian terdapat max pool layer dengan stride 2 juga. Convolution layer berikutnya terdapat 1 x 1, 64 kernel dan diikuti 3 x 3, 64 kernel, dan 1 x 1, 256 kernel. Ketiga layer ini diulangi sebanyak 3 kali sehingga memberikan 9 layer pada tahap ini. Berikutnya, terdapat kernel 1 x 1, 128 kernel, 3 x 3, 128 kernel, dan 1 x

1, 512 kernel sebanyak 4 kali, sehingga memberikan 12 layer pada tahap ini. Kemudian, terdapat 1 x 1, 256 kernel dan dua 3 x 3, 256 kernel, serta 1 x 1, 1024 kernel sebanyak 6 kali, sehingga memberikan 18 layer secara total pada tahap ini. Sekali lagi, terdapat 1 x 1, 512 kernel, dua 3 x 3, 512 kernel, dan 1 x 1, 2048 kernel diiulangi sebanyak 3 kali, sehingga memberikan 9 layer total pada tahap ini.

Setelah itu, dilakukan average pool dan diakhiri dengan fully connected layer yang mengandung 1000 node dan menggunakan activation fuction softmax sehingga memberikan 1 layer (tanpa menghitung activation fucntion dan max/average pool layer). Total terdapat 50 layer deep convolutional network.