Nama : Michael Christopher
NIM : 1103210260
Analisis UTS DL Soal #2

Penjelasan Fenomena dan Masalah dalam CNN

1. Fenomena Vanishing Gradient dan Mitigasi

Pada kasus CNN dengan beberapa lapisan konvolusi yang menghasilkan akurasi training 98% tetapi akurasi validasi hanya 62%, terjadi masalah overfitting yang parah. Salah satu penyebabnya bisa jadi adalah fenomena vanishing gradient.

Fenomena Vanishing Gradient pada Lapisan Awal:

Vanishing gradient terjadi ketika gradien yang digunakan untuk memperbarui bobot menjadi sangat kecil (mendekati nol) saat dibackpropagate ke lapisan-lapisan awal. Hal ini disebabkan oleh:

- Penggunaan fungsi aktivasi seperti sigmoid atau tanh yang memiliki rentang gradien terbatas
- Perkalian berulang gradien yang nilainya kurang dari 1 selama backpropagation
- Kedalaman jaringan yang tinggi (banyak lapisan)

Akibatnya, lapisan-lapisan awal pada CNN belajar sangat lambat atau bahkan tidak belajar sama sekali, sementara lapisan akhir mendominasi proses pembelajaran. Ini menyebabkan model tidak dapat mengekstrak fitur dasar yang baik dari data input.

Strategi Mitigasi Vanishing Gradient:

1. **Gunakan fungsi aktivasi ReLU atau variannya**: ReLU, Leaky ReLU, atau ELU memiliki gradien 1 untuk input positif, sehingga mengurangi masalah vanishing gradient.
2. **Implementasi skip connections**: Arsitektur seperti ResNet menggunakan skip connections yang memungkinkan gradien mengalir langsung ke lapisan sebelumnya tanpa teredam.
3. **Inisialisasi bobot yang tepat**: Gunakan metode inisialisasi seperti He initialization untuk ReLU atau Xavier/Glorot initialization untuk aktivasi lainnya.
4. **Normalisasi gradient**: Teknik seperti gradient clipping untuk mencegah gradien meledak atau menjadi terlalu kecil.
5. **Gunakan arsitektur yang lebih dangkal**: Kurangi jumlah lapisan jika memungkinkan atau gunakan arsitektur yang dirancang untuk jaringan dalam.

Masalah Batch Normalization:

Penambahan Batch Normalization setelah lapisan konvolusi ke-Y mungkin memperburuk generalisasi karena:

1. **Internal Covariate Shift dalam lapisan tengah**: Batch Normalization memang dirancang untuk mengatasi covariate shift, tetapi penempatan yang tidak tepat bisa mengganggu representasi fitur yang sudah terbentuk dengan baik.
2. **Noise pada mini-batch statistik**: Terutama jika ukuran batch kecil, statistik (mean dan variance) menjadi tidak stabil, menghasilkan normalisasi yang tidak konsisten.
3. **Overfitting terhadap statistik batch**: Model mungkin terlalu bergantung pada statistik normalisasi batch tertentu dan tidak generalisasi dengan baik.

Strategi Alternatif untuk Stabilisasi:

1. **Layer Normalization**: Tidak bergantung pada statistik batch sehingga lebih stabil untuk ukuran batch kecil.
2. **Group Normalization**: Membagi channel menjadi grup-grup dan menormalkan dalam setiap grup, mengurangi ketergantungan pada ukuran batch.
3. **Weight Standardization**: Mengaplikasikan normalisasi pada bobot, bukan pada aktivasi.
4. **Gunakan regularisasi lain**: Dropout, L1/L2 regularization, atau data augmentation untuk meningkatkan generalisasi.
5. **Pelatihan dengan curriculum learning**: Mulai dengan contoh mudah dan secara bertahap meningkatkan kesulitan.

2. Stagnasi Loss Training pada CNN

Ketika melatih CNN dari nol dan loss training stagnan di nilai tinggi setelah ratusan epoch, ada beberapa penyebab potensial yang perlu dipertimbangkan:

Tiga Penyebab Potensial Stagnasi:

1. **Masalah Learning Rate:**
    - **Learning rate terlalu besar**: Menyebabkan osilasi dan gagal konvergen
    - **Learning rate terlalu kecil**: Kemajuan sangat lambat atau terjebak di saddle point
    - **Learning rate decay tidak tepat**: Penurunan learning rate terlalu cepat atau terlalu lambat
2. **Masalah Inisialisasi Bobot:**
    - **Vanishing/exploding gradients** akibat inisialisasi yang buruk
    - **Dead neurons** saat awal pelatihan karena inisialisasi yang tidak sesuai dengan fungsi aktivasi
    - **Symmetry breaking** yang tidak memadai, menyebabkan neuron-neuron belajar fitur yang sama
3. **Masalah Kompleksitas Model:**
    - **Model terlalu sederhana** (underfitting) untuk menangkap kompleksitas data
    - **Arsitektur tidak sesuai** dengan karakteristik permasalahan
    - **Bottleneck** yang terlalu sempit menyebabkan hilangnya informasi penting

Cyclic Learning Rate dan Local Minima:

Cyclic Learning Rate (CLR) dapat membantu model keluar dari local minima karena:

1. **Variasi sistematis learning rate**: CLR bergerak antara nilai minimum dan maksimum secara periodik, memungkinkan model melewati local minima saat learning rate tinggi.
2. **Eksplorasi landscape loss function**: Saat learning rate tinggi, model dapat melompati hambatan energi dan menjelajahi area baru dari ruang parameter.
3. **Annealing effect**: Penurunan learning rate selama siklus memungkinkan model untuk konvergen ke minima yang lebih baik setelah melewati local minima.
4. **Ensemble-like effect**: Berbagai learning rate menghasilkan solusi yang berbeda yang kemudian digabungkan saat model dilatih lebih lanjut.

Pengaruh Momentum pada Optimizer SGD:

Momentum pada SGD mempengaruhi konvergensi dengan cara:

1. **Mempercepat konvergensi**: Momentum menambahkan fraksi update sebelumnya ke update saat ini, mempercepat gerakan di arah yang konsisten.
2. **Membantu melewati local minima**: Akumulasi momentum memungkinkan optimizer melompati local minima yang dangkal.
3. **Meredam osilasi**: Momentum meredam pergerakan yang berlawanan arah, menghaluskan trajectory optimization.
4. **Meningkatkan stabilitas**: Membantu mengatasi noise dalam gradien dengan mengambil rata-rata bergerak dari update.
5. **Adaptive behavior**: Momentum tinggi (0.9-0.99) memungkinkan adaptasi lebih cepat pada arah gradien yang konsisten namun bisa menyebabkan overshoot jika terlalu tinggi.

3. Fenomena Dying ReLU dalam CNN

Pada kasus klasifikasi spesies ikan menggunakan CNN dengan fungsi aktivasi ReLU yang tidak menunjukkan peningkatan akurasi setelah 50 epoch meskipun learning rate telah dioptimasi, kemungkinan besar terjadi fenomena dying ReLU.

Fenomena Dying ReLU:

Dying ReLU terjadi ketika neuron dengan aktivasi ReLU secara konsisten menghasilkan output nol untuk semua input, efektif "mati" dan tidak lagi berpartisipasi dalam proses pembelajaran. Ini terjadi karena:

1. **ReLU hanya meloloskan nilai positif**: ReLU(x) = max(0, x), berarti semua input negatif diubah menjadi nol.
2. **Gradien nol untuk input negatif**: Ketika input negatif, gradien ReLU adalah nol, sehingga tidak ada update bobot yang terjadi saat backpropagation.
3. **Siklus negatif yang merugikan**: Jika bobot diperbarui sedemikian rupa sehingga neuron selalu menerima input negatif, neuron tersebut tidak akan pernah diaktifkan lagi dan tidak akan pernah diperbarui.

Dampak Dying ReLU pada Backpropagation:

Fenomena dying ReLU mengganggu aliran gradien selama backpropagation dengan cara:

1. **Memblokir aliran informasi**: Neuron yang mati tidak meneruskan sinyal atau gradien, menciptakan "bottleneck" dalam jaringan.
2. **Mengurangi kapasitas model**: Semakin banyak neuron yang mati, semakin sedikit kapasitas model untuk belajar dan beradaptasi.
3. **Mengganggu representasi fitur**: Neuron yang mati tidak berkontribusi pada ekstraksi fitur, menghasilkan representasi yang kurang kaya.
4. **Memperlambat atau menghentikan pembelajaran**: Dengan banyak neuron yang mati, model mungkin tidak dapat belajar lebih lanjut meskipun learning rate telah dioptimasi.
5. **Pembaruan bobot tidak seimbang**: Neuron aktif mendapatkan lebih banyak pembaruan, sementara neuron mati tetap tidak berubah.

Solusi untuk Dying ReLU:

1. **Gunakan varian ReLU lain**:
    - Leaky ReLU: $f(x) = \max(\alpha x, x)$, α kecil (misalnya 0.01)
    - Parametric ReLU (PReLU): α dipelajari selama pelatihan
    - Exponential Linear Unit (ELU): memberikan output negatif kecil untuk input negatif
    - Randomized Leaky ReLU (RReLU): α dipilih secara acak
2. **Inisialisasi bobot yang tepat**: Menggunakan He initialization yang dirancang khusus untuk ReLU
3. **Penyesuaian learning rate**: Learning rate yang lebih kecil dapat mengurangi risiko neuron mati
4. **Batch Normalization**: Membantu menjaga distribusi aktivasi lebih stabil
5. **Residual connections**: Memungkinkan sinyal melewati neuron yang mati melalui shortcut connections

4. Masalah Kinerja Kelas Spesifik dalam Klasifikasi Ikan

Pada kasus klasifikasi ikan dengan grafik AUC-ROC yang menunjukkan satu kelas (Spesies X) stagnan di 0.55 sementara kelas lain mencapai >0.85, kita perlu memahami mengapa class-weighted loss function gagal meningkatkan kinerja spesies tersebut.

Alasan Kegagalan Class-weighted Loss Function:

1. **Heterogenitas dalam kelas**: Spesies X mungkin memiliki variasi visual yang sangat besar dibandingkan spesies lain, sehingga hanya memberikan bobot lebih tidak mencukupi.
2. **Kualitas data, bukan kuantitas**: Meskipun dengan pembobotan, jika sampel Spesies X memiliki kualitas rendah (blurry, terhalang, sudut yang tidak biasa), kinerja tetap buruk.
3. **Overlap fitur dengan kelas lain**: Spesies X mungkin berbagi karakteristik visual yang sangat mirip dengan spesies lain, sehingga pembobotan tidak menyelesaikan masalah kebingungan klasifikasi.

Tiga Faktor Penyebab Potensial:

1. **Terkait Karakteristik Data**:
    - **Variabilitas intrinsik**: Spesies X mungkin memiliki variabilitas fenotipik tinggi (warna, ukuran, pola) yang menyulitkan pembelajaran fitur diskriminatif.

    - **Noise dan outlier**: Sampel Spesies X mungkin mengandung lebih banyak noise atau outlier yang mengganggu pembelajaran.
    - **Bias pengambilan sampel**: Kondisi pengambilan gambar Spesies X mungkin sangat berbeda (pencahayaan, sudut, lingkungan) dibandingkan spesies lain.
2. **Terkait Arsitektur Model**:
    - **Receptive field tidak tepat**: Ukuran filter konvolusi mungkin tidak optimal untuk menangkap fitur diskriminatif Spesies X.
    - **Kapasitas representasi tidak mencukupi**: Jumlah filter atau lapisan mungkin tidak cukup untuk mengekstrak fitur halus yang membedakan Spesies X.
    - **Feature map mismatch**: Arsitektur mungkin menghasilkan feature map yang terlalu kecil untuk mendeteksi pola detail pada Spesies X.
3. **Terkait Optimasi dan Pembelajaran**:
    - **Pembobotan yang tidak optimal**: Class weighting mungkin terlalu ekstrem (terlalu tinggi/rendah), menyebabkan ketidakstabilan dalam pembelajaran.
    - **Focal loss mungkin lebih tepat**: Kasus ini mungkin memerlukan focal loss yang fokus pada sampel sulit daripada hanya class weighting.
    - **Learning rate tidak sesuai**: Learning rate mungkin tidak optimal untuk kelas dengan sampel yang lebih menantang.

Solusi Potensial:

1. **Analisis kelas bermasalah**:
    - Visualisasi sampel Spesies X untuk memahami heterogenitas dan tantangannya
    - Gunakan teknik seperti t-SNE untuk melihat overlapping dengan kelas lain
2. **Pendekatan loss function alternatif**:
    - Gunakan focal loss: $\alpha(1-p)^{\gamma} \log(p)$ yang memberikan lebih banyak bobot pada sampel sulit
    - Implementasikan hard negative mining untuk fokus pada sampel yang sering salah klasifikasi
3. **Modifikasi arsitektur**:
    - Tambahkan cabang khusus untuk Spesies X yang memiliki lebih banyak filter
    - Eksperimen dengan filter size berbeda untuk menangkap fitur spesifik Spesies X
4. **Data augmentation spesifik**:
    - Terapkan augmentasi lebih agresif pada Spesies X
    - Gunakan teknik seperti mixup atau CutMix khusus untuk kelas yang bermasalah

5. Overfitting dan Kesalahan Desain Arsitektur CNN

Pada kasus arsitektur CNN untuk klasifikasi ikan di mana peningkatan kompleksitas model justru menyebabkan penurunan akurasi validasi dari 85% ke 65% (meskipun akurasi training mencapai 98%), kita melihat fenomena overfitting yang klasik.

Fenomena Overfitting:

Overfitting terjadi ketika model terlalu "menghafalkan" data training, sehingga menghasilkan performa yang sangat baik pada data training tetapi buruk pada data validasi/testing. Ini terjadi karena:

1. **Model mempelajari noise**: Selain pola yang valid, model juga mempelajari noise dan keunikan spesifik yang hanya ada di data training.
2. **Hilangnya kemampuan generalisasi**: Model menjadi terlalu spesifik terhadap data training dan kehilangan kemampuan untuk menggeneralisasi ke data baru.
3. **Kompleksitas berlebihan**: Model memiliki terlalu banyak parameter dibandingkan dengan jumlah sampel training yang tersedia.

Mengapa Penambahan Kapasitas Tidak Selalu Meningkatkan Generalisasi:

1. **Bias-variance tradeoff**: Peningkatan kompleksitas menurunkan bias (kemampuan mengepaskan data training) tetapi meningkatkan variance (sensitivitas terhadap fluktuasi dalam data training).
2. **Jumlah data terbatas**: Tanpa peningkatan jumlah data training, penambahan kapasitas model hanya membuat model lebih mudah menghafalkan daripada belajar pola yang bermakna.
3. **Curse of dimensionality**: Semakin tinggi dimensi ruang fitur, semakin banyak data yang dibutuhkan untuk mengisi ruang tersebut secara bermakna.
4. **Peningkatan noise learning**: Model yang lebih kompleks memiliki lebih banyak parameter untuk menyesuaikan dengan noise di data training.

Tiga Kesalahan Desain Arsitektur yang Memicu Degradasi Performa:

1. **Kedalaman yang berlebihan tanpa residual connections**:
    - Terlalu banyak lapisan konvolusional tanpa residual connections menyebabkan vanishing gradients
    - Solusi: Tambahkan skip connections (ResNet style) atau kurangi jumlah lapisan
2. **Rasio upsampling/downsampling yang tidak tepat**:
    - Terlalu banyak downsampling (pooling berlebihan) menyebabkan hilangnya informasi spasial penting
    - Downsampling yang tidak cukup menyebabkan parameter berlebihan dan komputasi berat
    - Solusi: Seimbangkan downsampling dengan kebutuhan informasi spasial untuk klasifikasi ikan
3. **Distribusi filter yang tidak optimal**:
    - Terlalu banyak filter di lapisan awal dan terlalu sedikit di lapisan dalam, atau sebaliknya
    - Arsitektur "bottleneck" yang terlalu sempit menyebabkan hilangnya informasi
    - Solusi: Distribusikan filter secara proporsional, dengan jumlah filter yang meningkat secara bertahap dari lapisan awal ke lapisan dalam

Strategi Mengatasi Overfitting:

1. **Regularisasi yang tepat**:
    - Dropout: Matikan neuron secara acak selama pelatihan
    - Weight decay (L1/L2 regularization): Penalti untuk bobot besar
    - Early stopping: Berhenti melatih ketika performa validasi mulai menurun
2. **Data augmentation**:
    - Transformasi pada data training: rotasi, flipping, zooming, dll.

    - Mixup atau CutMix: Mencampur dua sampel training untuk meningkatkan generalisasi
3. **Arsitektur yang lebih sederhana dan efisien**:
    - MobileNet atau EfficientNet style: Arsitektur yang dirancang untuk efisiensi
    - Pruning: Memangkas koneksi yang tidak penting
    - Knowledge distillation: Melatih model yang lebih kecil untuk menyamai performa model besar
4. **Ensemble models**:
    - Gabungkan beberapa model yang lebih sederhana daripada satu model yang sangat kompleks

Dengan mengenali masalah-masalah ini dan menerapkan solusi yang tepat, Anda dapat membangun CNN yang memiliki performa dan generalisasi yang lebih baik untuk klasifikasi ikan.