LAPORAN UAS MACHINE LEARNING

Technical Report Pytorch Tutorial

Oleh : April Hamonangan Marbun
NIM : 1103202039

PRODI S1 TEKNIK KOMPUTER
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
2023

1. Chapter 00 Pytorch Fundamental

PyTorch, sebuah framework machine learning dan deep learning sumber terbuka, menjadi pusat perhatian dari laporan teknis komprehensif ini. PyTorch dikenal luas karena fleksibilitasnya, grafik komputasi dinamis, dan integrasinya yang mulus dengan GPU. Laporan ini bertujuan untuk menjelaskan konsep dan fungsionalitas inti PyTorch, mengakomodasi pemula dan praktisi di bidang deep learning. PyTorch digunakan secara luas dalam berbagai aplikasi, mulai dari tugas machine learning tradisional hingga model deep learning yang kompleks. Kelebihannya memungkinkan praktisi untuk mengimplementasikan berbagai algoritma, menjadikannya cocok untuk tugas seperti pengenalan gambar dan suara, pemrosesan bahasa alami, dan pembelajaran penguatan. Sifat dinamis grafik komputasi PyTorch memberdayakan peneliti dan pengembang untuk bereksperimen dengan arsitektur model secara efisien.

Pengguna terkemuka PyTorch melibatkan peneliti, ilmuwan data, dan insinyur di berbagai bidang akademis dan industri. Popularitasnya dikarenakan sintaksis yang intuitif, dokumentasi yang ekstensif, dan dukungan dari komunitas yang dinamis. PyTorch seringkali menjadi pilihan utama bagi mereka yang menghargai komputasi grafik dinamis, fitur kunci dalam lingkungan eksperimental dan penelitian. Keputusan untuk menggunakan PyTorch dibenarkan oleh antarmuka yang ramah pengguna, dokumentasi yang kokoh, dan dukungan untuk grafik komputasi dinamis. Peneliti terutama menghargai kemudahan debugging dan eksperimen, menjadikannya alat pilihan untuk prototyping dan pengembangan model baru. Selain itu, integrasi yang mulus dengan GPU memungkinkan pelatihan yang efisien dari jaringan saraf berskala besar, berkontribusi pada perhitungan yang lebih cepat dan terobosan dalam penelitian deep learning.

Modul ini mencakup berbagai topik, dimulai dari dasar-dasar mengimpor PyTorch dan pengenalan terhadap tensor. Laporan ini membahas pembuatan tensor, mengeksplorasi random tensors, zeros, ones, dan tensors dengan rentang tertentu. Detail jenis data tensor, mendapatkan informasi dari tensor, dan memanipulasinya melalui operasi seperti mereshape, stacking, squeezing, dan unsqueezing. Keberhasilan perkalian matriks diutarakan, menyoroti perannya yang sentral dalam operasi deep learning. Laporan ini mengatasi kesalahan umum, terutama masalah terkait bentuk, yang mendasar untuk mencegah inkonsistensi selama pelatihan model. Operasi agregasi, seperti menemukan nilai minimum, maksimum, rata-rata, dan jumlah, dibahas secara rinci. Konsep min/max posisional memberikan wawasan dalam mengekstrak elemen-elemen spesifik dari tensor. Laporan juga membimbing pengguna dalam mengubah jenis data tensor dan menjelajahi teknik perubahan bentuk data.

Pembahasan mendalam diberikan pada pengindeksan, menampilkan bagaimana memilih data dari tensor dengan efektif. Integrasi antara PyTorch dan NumPy dijelajahi, menekankan interoperabilitas kedua perpustakaan yang kuat ini. Reproducibility menjadi pertimbangan kunci dalam deep learning, dan laporan ini membahas strategi untuk mencapainya. Teknik pengelolaan acak dibahas, mengatasi tantangan membuat eksperimen lebih deterministik. Modul ini ditutup dengan panduan komprehensif untuk menjalankan tensor di GPU. Langkah-langkah untuk mendapatkan GPU, mengonfigurasi PyTorch untuk penggunaan GPU, dan memindahkan tensor dengan efisien antara CPU dan GPU diuraikan. Pengetahuan ini sangat penting untuk memanfaatkan kekuatan komputasi penuh GPU, menghasilkan alur kerja deep learning yang lebih cepat dan efisien.

2. Chapter 01 PyTorch Workflow Fundamentals

Dalam tahap awal pengembangan model PyTorch, langkah penting pertama adalah menyiapkan dan memuat dataset. Proses ini mencakup pembagian data ke dalam set pelatihan dan uji, membentuk landasan evaluasi model yang efektif. Selanjutnya, pada tahap pembangunan model, kita memahami esensi dari komponen-komponen penting dalam PyTorch. Hal ini mencakup pemeriksaan isi model PyTorch dan pemahaman cara membuat prediksi menggunakan torch.inference_mode().

Langkah selanjutnya adalah pelatihan model, dimana inti dari deep learning terletak. Dalam bagian ini, kita belajar membuat fungsi kerugian dan pengoptimal di PyTorch, yang sangat penting untuk mengoptimalkan parameter model. Loop optimisasi diperkenalkan, yang mencakup kedua loop pelatihan dan pengujian, memastikan evaluasi kinerja model dengan cermat. Setelah model dilatih, langkah berikutnya adalah fase inferensi, di mana model diuji dengan data baru yang belum pernah dilihat sebelumnya.

Pentingnya menyimpan dan memuat model dalam PyTorch juga dibahas. Kita menjelajahi cara menyimpan state_dict() model PyTorch dan memuat state_dict() model yang telah disimpan, memastikan pelestarian parameter yang telah dipelajari untuk penggunaan masa depan. Lalu, kita menyatukan semua komponen dalam contoh praktis, memulai dari persiapan dan pemuatan data, pembangunan model linear PyTorch, pelatihan, membuat prediksi pada data baru, hingga menyimpan dan memuat model. Keseluruhan alur kerja pengembangan model PyTorch dieksplorasi secara rinci, memberikan dasar untuk aplikasi yang lebih lanjut pada bab-bab berikutnya.

3. Chapter 02 PyTorch Neural Network Classification

Arsitektur jaringan saraf untuk klasifikasi melibatkan serangkaian langkah, dimulai dari persiapan data hingga evaluasi model. Pada tahap pertama, kita membuat data klasifikasi dan menyiapkannya untuk pelatihan. Ini mencakup pembentukan bentuk input dan output, mengonversi data menjadi tensor, dan pembagian data menjadi set pelatihan dan uji. Selanjutnya, kita membangun model klasifikasi. Langkah ini melibatkan penentuan fungsi kerugian dan optimizer yang akan digunakan selama pelatihan. Proses pelatihan model diperinci, termasuk transformasi output model menjadi label prediksi.

Setelah model terlatih, kita melakukan evaluasi dengan membuat prediksi pada data uji dan mengevaluasi kinerja model. Pada tahap ini, kita dapat memperbaiki model dari perspektif arsitektur untuk memastikan bahwa model dapat mencocokkan pola data tertentu. Eksplorasi tambahan dilakukan dengan mengintegrasikan unsur non-linearitas pada model. Data non-linear direplikasi untuk menunjukkan pentingnya penggunaan fungsi aktivasi non-linear dalam jaringan saraf.

Langkah selanjutnya melibatkan pembangunan model klasifikasi multi-kelas di PyTorch. Proses ini mencakup pembuatan data klasifikasi multi-kelas, pembentukan model, dan penyiapan fungsi kerugian serta optimizer untuk model tersebut. Evaluasi model multi-kelas juga diperinci untuk memahami performanya.

Terakhir, laporan mencakup evaluasi model klasifikasi dengan berbagai metrik. Ini melibatkan perhitungan akurasi, presisi, recall, dan F1-score, memberikan pemahaman holistik tentang kinerja model dalam tugas klasifikasi. Seluruh laporan bertujuan untuk memberikan panduan komprehensif dalam pengembangan dan evaluasi model klasifikasi jaringan saraf.

4. Chapter 03 PyTorch Computer Vision

Eksplorasi pengembangan dan pelatihan model computer vision menggunakan framework PyTorch dimulai dengan langkah penting dalam mendapatkan dataset yang sesuai. Pemahaman terhadap bentuk input dan output dari model computer vision menjadi kunci, dan laporan ini dengan cermat menjelaskan aspek-aspek tersebut untuk membimbing pengembangan model selanjutnya. Visualisasi dataset ditekankan sebagai langkah awal yang penting, memberikan wawasan berharga terhadap sifat data, yang kemudian menjadi dasar keputusan pengembangan model.

Selanjutnya, kami memasuki tahap persiapan DataLoader, komponen penting untuk menangani data secara efisien dalam bentuk batch selama proses pelatihan. Kemudian, kami membangun Model 0, yang berfungsi sebagai model dasar. Bagian ini melibatkan penyiapan fungsi loss, optimizer, dan metrik evaluasi. Selain itu, kami memperkenalkan sebuah fungsi untuk mengukur waktu eksperimen kami.

Pada tahap selanjutnya, laporan ini mengeksplorasi pembangunan Model 1 dengan memperkenalkan non-linearitas. Fungsi loss, optimizer, dan metrik evaluasi juga disiapkan. Proses pelatihan dan pengujian model dijabarkan ke dalam fungsi yang mempermudah.

Langkah berikutnya adalah membangun Model 2, yakni Convolutional Neural Network (CNN). Pemahaman terhadap modul nn.Conv2d() dan nn.MaxPool2d() disajikan secara rinci. Fungsi loss dan optimizer khusus untuk Model 2 diperkenalkan, diikuti dengan proses pelatihan dan pengujian menggunakan fungsi yang telah disiapkan sebelumnya. Dalam mengambil keputusan tentang model yang akan digunakan, laporan ini membahas trade-off antara performa dan kecepatan. Penyusunan prediksi acak dengan model terbaik, pembuatan matriks kebingungan, dan penyimpanan serta pemanggilan model terbaik juga dijelaskan secara komprehensif.

5. Chapter 04 PyTorch Custom Datasets

Dalam laporan teknis ini, kami menjelajahi langkah-langkah dalam membangun model klasifikasi gambar menggunakan framework PyTorch. Bab pertama dimulai dengan mengimpor PyTorch dan menyiapkan kode yang dapat beradaptasi dengan perangkat, memastikan pelaksanaan yang lancar baik pada CPU maupun GPU. Langkah selanjutnya adalah memperoleh data yang diperlukan untuk klasifikasi gambar. Dua opsi dijelaskan, yakni memuat data gambar menggunakan ImageFolder dan memuat data gambar dengan dataset kustom.

Seiring dengan itu, persiapan data menjadi tahap penting dalam melatih model. Visualisasi gambar digunakan untuk memperoleh wawasan tentang dataset yang akan digunakan. Transformasi data menjadi fokus pada bab ketiga, dengan menggunakan modul torchvision.transforms untuk memproses gambar sebelum diberikan ke dalam model. Opsi untuk memuat data menggunakan ImageFolder dan dataset kustom diuraikan dalam bab empat dan lima.

Bab keenam memperkenalkan model pertama, TinyVGG, tanpa augmentasi data. Proses ini melibatkan pembuatan transformasi dan memuat data untuk Model 0, serta pembuatan fungsi pelatihan dan evaluasi. Selanjutnya, bab tujuh memperkenalkan konsep kurva kehilangan ideal, overfitting, dan underfitting.

Model kedua, TinyVGG dengan augmentasi data, diperkenalkan pada bab sembilan. Augmentasi data

dijelaskan melalui transformasi kustom untuk meningkatkan keberagaman data. Pembuatan dan pelatihan Model 1 juga diperincikan dalam bab ini. Terakhir, bab sebelas menunjukkan cara membuat prediksi pada gambar kustom menggunakan model yang telah dilatih sebelumnya. Proses ini melibatkan pemuatan dan prediksi pada gambar kustom dengan menggunakan PyTorch. Keseluruhan laporan teknis ini memberikan panduan langkah-demi-langkah dalam mengimplementasikan model klasifikasi gambar dengan PyTorch.

6. Chapter 05 PyTorch Going Modular

Dalam laporan teknis ini, kami memulai dengan membahas perbedaan antara mode sel dan mode skrip dalam lingkungan PyTorch. Bab pertama memberikan pemahaman tentang karakteristik masing-masing mode, yaitu mode sel untuk eksekusi per sel dan mode skrip untuk menjalankan seluruh skrip secara utuh. Langkah berikutnya adalah memperoleh data, yang merupakan langkah kritis dalam proses pengembangan model. Bab kedua fokus pada pembuatan dataset dan dataloader menggunakan skrip data_setup.py. Pendekatan ini memungkinkan keterpisahan dan pengorganisasian yang efisien terhadap manipulasi data.

Selanjutnya, pembuatan model dibahas dalam bab ketiga, dijelaskan melalui skrip model_builder.py. Penekanan diberikan pada kejelasan struktur dan konfigurasi model, memudahkan pengelolaan dan pemeliharaan model dalam pengembangan yang lebih kompleks.

Implementasi langkah-langkah pelatihan dan evaluasi model terdapat pada bab keempat. Fungsi train_step() dan test_step() dirancang untuk memudahkan pelatihan dan pengujian model. Fungsi train() digunakan untuk menggabungkan kedua langkah tersebut, memastikan proses pelatihan dan pengujian dapat dilakukan secara bersamaan. Bab kelima menguraikan pembuatan fungsi untuk menyimpan model, yang ditempatkan dalam skrip utils.py. Hal ini memastikan model yang telah dilatih dapat disimpan dengan efisien untuk penggunaan atau evaluasi lebih lanjut.

Terakhir, bab keenam membahas proses lengkap pelatihan, evaluasi, dan penyimpanan model dalam skrip train.py. Keseluruhan laporan teknis ini memberikan panduan komprehensif tentang proses pengembangan model dengan memanfaatkan seluk-beluk PyTorch dan praktik terbaik dalam organisasi dan struktur kode.

7. Chapter 06 PyTorch Transfer Learning

Chapter 06 proyek transfer learning dengan PyTorch dimulai dengan menetapkan lingkungan kerja dan mengimpor pustaka yang diperlukan. Proses ini mencakup instalasi PyTorch dan torchvision, dengan fokus pada memastikan fleksibilitas perangkat keras melalui kode yang dapat beradaptasi dengan perangkat (device-agnostic). Langkah berikutnya adalah mendapatkan dataset, langkah kritis dalam transfer learning. Proyek ini memberikan opsi untuk menggunakan dataset kustom atau dataset umum dari torchvision, yang membentuk dasar pelatihan dan pengujian model.

Dalam pembahasan selanjutnya, transformasi data didefinisikan untuk mempersiapkan data. Ini mencakup pembuatan transformasi secara manual dan otomatis untuk menyesuaikan dengan model-model torchvision. Transformasi ini memainkan peran kunci dalam memastikan data siap untuk pelatihan. Langkah berikutnya adalah mendapatkan model yang telah dilatih sebelumnya. Proyek ini memberikan informasi tentang model yang dapat dipilih dan memberikan langkah-langkah untuk menyiapkan model tersebut untuk tugas spesifik.

Proses pelatihan model melibatkan pembekuan bagian tertentu dari model dan penyesuaian output layer sesuai kebutuhan tugas. Selanjutnya, model dievaluasi dengan memplot kurva kerugian (loss curves) untuk memahami kinerjanya.

Setelah pelatihan, model diuji pada gambar-gambar dari set pengujian untuk membuat prediksi. Langkah ini memberikan pemahaman tentang sejauh mana model dapat membuat prediksi yang akurat pada data yang belum pernah dilihat. Chapter ini diakhiri dengan langkah-langkah untuk membuat prediksi pada gambar-gambar kustom dan mengevaluasi performa model pada tugas tersebut.

8. Chapter 07 PyTorch Experiment Tracking

Bab 07 difokuskan pada pelacakan eksperimen menggunakan PyTorch dan TensorBoard. Langkah awal melibatkan penyiapan lingkungan dengan membuat fungsi bantu untuk menetapkan seed, sehingga dapat memastikan reproduktibilitas eksperimen. Bagian-bagian berikutnya membahas pengambilan dan persiapan data untuk eksperimen. Dua pendekatan disajikan untuk membuat DataLoaders. Pendekatan pertama melibatkan pembuatan transformasi secara manual, sementara pendekatan kedua menggunakan transformasi yang dibuat secara otomatis. Tujuannya adalah memfasilitasi eksperimen dengan berbagai metode pemrosesan data.

Inti dari bab ini berkisar pada penggunaan model yang telah dilatih sebelumnya, khususnya membekukan lapisan dasar dan mengadaptasi bagian klasifikasinya. Pelatihan model diperkaya dengan pelacakan hasil menggunakan fungsi SummaryWriter(). Fungsi train() diperluas untuk menggabungkan mekanisme pelacakan ini.

Untuk memberikan representasi visual dari kinerja model, TensorBoard diperkenalkan. Fungsi bantu dibuat untuk membangun instansi SummaryWriter(), dan fungsi train() diperbarui untuk memasukkan parameter penulis agar integrasinya menjadi lebih lancar.

Bab ini juga menguraikan konsep melakukan serangkaian eksperimen pemodelan. Ini menyoroti pentingnya menentukan jenis eksperimen yang akan dijalankan dan menguraikan jenis eksperimen yang akan dilakukan. Selain itu, bab ini membahas unduhan dataset yang berbeda, transformasi dataset, serta pembuatan model ekstraktor fitur dan eksperimen, lengkap dengan kode pelatihan. Akhirnya, bab ini mengeksplorasi cara melihat hasil eksperimen menggunakan TensorBoard.