**Melatih dan mengevaluasi model pembelajaran mendalam**

Pembelajaran dalam adalah bentuk pembelajaran mesin tingkat lanjut yang meniru cara belajar otak manusia melalui jaringan neuron yang terhubung.

**Tujuan pembelajaran**

Dalam modul ini, Anda akan mempelajari:

* Prinsip-prinsip dasar pembelajaran dalam
* Cara melatih jaringan neural dalam (DNN) menggunakan PyTorch atau Tensorflow
* Cara melatih jaringan neural konvolusional (CNN) menggunakan PyTorch atau Tensorflow
* Cara menggunakan pemelajaran transfer untuk melatih jaringan neural konvolusional (CNN) dengan PyTorch atau Tensorflow

1. **Pengantar**

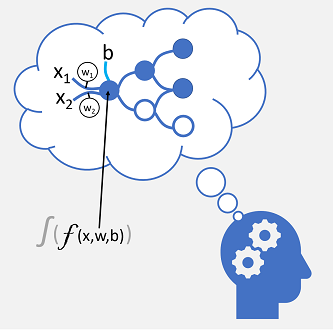
Pembelajaran dalam adalah bentuk pembelajaran mesin tingkat lanjut yang mencoba meniru cara belajar otak manusia.

Di otak Anda, ada sel-sel neural yang disebut neuron yang saling terhubung oleh ekstensi neural yang melewati sinyal elektrokimia melalui jaringan.



Ketika neuron pertama dalam jaringan dirangsang, sinyal input akan diproses, dan jika melebihi ambang tertentu, neuron akan diaktifkan serta meneruskan sinyal ke neuron yang terhubung dengan neuron pertama. Neuron ini selanjutnya dapat diaktifkan dan meneruskan sinyal ke seluruh jaringan yang tersisa. Dari waktu ke waktu, koneksi antara neuron akan semakin kuat karena sering digunakan saat Anda terus mempelajari cara menanggapi secara efektif. Misalnya, jika seseorang melempar bola ke arah Anda, koneksi neuron memungkinkan Anda untuk memproses informasi visual dan mengoordinasikan gerakan untuk menangkap bola. Jika Anda melakukan tindakan ini berulang kali, jaringan neuron yang terlibat untuk menangkap bola akan bertambah kuat saat Anda terus mempelajari cara menangkap bola dengan lebih baik.

Pembelajaran dalam meniru proses biologis tersebut dengan menggunakan jaringan neural tiruan yang memproses input numerik, bukan rangsangan elektrokimia.



Koneksi saraf yang masuk digantikan oleh input numerik yang biasanya diidentifikasi sebagai x. Ketika ada lebih dari satu nilai input, x dianggap sebagai vektor dengan elemen bernama x1, x2, dan sebagainya.

Yang dikaitkan dengan setiap nilai x adalah weight (w), yang digunakan untuk memperkuat atau melemahkan efek dari nilai x untuk menyimulasikan pembelajaran. Selain itu, input bias (b) ditambahkan untuk memungkinkan kontrol mendetail atas jaringan. Selama proses pelatihan, nilai w dan b akan disesuaikan untuk menyesuaikan jaringan sehingga jaringan "belajar" untuk menghasilkan output yang benar.

Neuron itu sendiri merangkum fungsi yang menghitung jumlah tertimbang dari x, w, dan b. Fungsi ini selanjutnya disertakan dalam activation function yang membatasi hasil (sering kali antara nilai 0 dan 1) untuk menentukan apakah neuron mengirimkan output ke lapisan neuron berikutnya yang ada di dalam jaringan.

1. **Konsep jaringan neural dalam**

Sebelum menjelajahi cara melatih model pembelajaran mesin jaringan neural dalam (DNN), mari mempertimbangkan apa yang ingin kita capai. Pembelajaran mesin berfokus untuk memprediksi label berdasarkan beberapa fitur dari pengamatan tertentu. Istilah sederhananya, model pembelajaran mesin adalah fungsi yang menghitung y (label) dari x (fitur): f(x)=y.

1. **Contoh klasifikasi sederhana**

Misalnya, anggaplah pengamatan Anda terdiri dari beberapa pengukuran seekor penguin



Secara khusus, pengukurannya adalah:

* Panjang paruh penguin.
* Kedalaman paruh penguin.
* Panjang sirip penguin.
* Bobot penguin.

Dalam hal ini, fitur (x) adalah vektor empat nilai, atau secara matematis, x=[x 1,x2,x3,x4].

Katakanlah label yang akan kita coba prediksi (y) adalah spesies penguin, dan ada tiga kemungkinan spesies yaitu:

* Adelie
* Gentoo
* Chinstrap

Ini adalah contoh masalah klasifikasi, di mana model pembelajaran mesin harus memprediksi ke kelas mana pengamatan akan dimasukkan, yang paling memungkinkan. Model klasifikasi menyelesaikan masalah tersebut dengan memprediksi label yang terdiri dari peluang untuk setiap kelas. Dengan kata lain, y adalah vektor dari tiga nilai probabilitas; satu untuk setiap kemungkinan kelas: y=[P(0),P(1),P(2)].

Anda melatih model pembelajaran mesin dengan menggunakan pengamatan yang sudah Anda ketahui label sebenarnya. Misalnya, Anda mungkin memiliki pengukuran fitur untuk spesimen Adelie sebagai berikut:

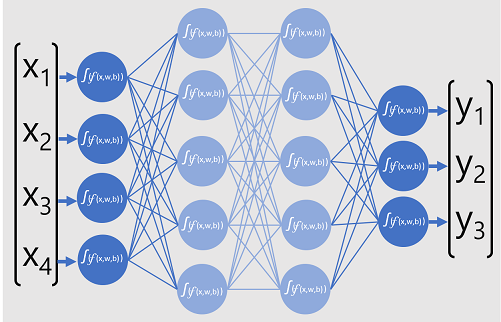
x=[37.3, 16.8, 19.2, 30.0]

Anda sudah tahu bahwa ini adalah contoh dari Adelie (kelas 0), maka fungsi klasifikasi yang sempurna harus menghasilkan label yang menunjukkan peluang 100% untuk kelas 0, dan peluang 0% untuk kelas 1 dan 2:

y=[1, 0, 0]

1. **Model jaringan neural dalam**

Jadi bagaimana kita akan menggunakan pembelajaran dalam untuk membangun model klasifikasi untuk model klasifikasi penguin? Lihatlah contoh berikut:



Model jaringan neural dalam untuk pengklasifikasi terdiri dari beberapa lapisan neuron tiruan. Dalam hal ini, ada empat lapisan yaitu:

* Lapisan input dengan neuron untuk setiap nilai input yang diharapkan (x).
* Dua lapisan tersembunyi, masing-masing berisi lima neuron.
* Lapisan output yang berisi tiga neuron - satu untuk setiap nilai probabilitas kelas (y) yang akan diprediksi oleh model.

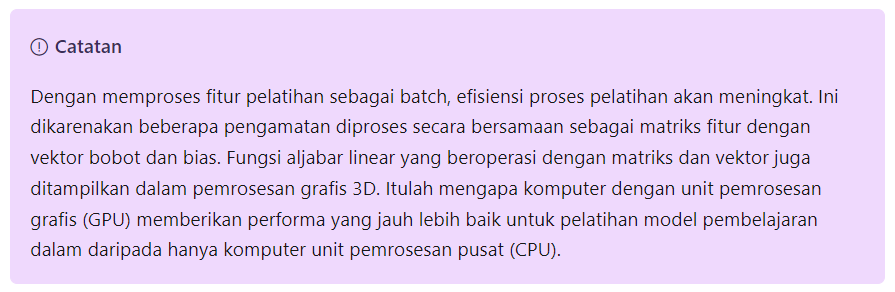
Dikarenakan arsitektur jaringan berlapis, model semacam ini terkadang disebut sebagai perceptron multilapisan. Selain itu, perhatikan bahwa semua neuron yang ada dalam lapisan input dan lapisan tersembunyi terhubung ke semua neuron di lapisan berikutnya - ini adalah contoh jaringan yang terhubung sepenuhnya.

Ketika membuat model seperti ini, Anda harus menentukan lapisan input yang mendukung jumlah fitur yang akan diproses model Anda dan lapisan output yang mencerminkan jumlah output yang Anda harapkan untuk dihasilkan. Anda dapat memutuskan berapa banyak lapisan tersembunyi yang ingin Anda sertakan dan berapa banyak neuron di masing-masing lapisan; tetapi Anda tidak punya kontrol atas nilai input dan output untuk lapisan ini - nilai input dan output ditentukan oleh proses pelatihan model.

1. **Melatih jaringan neural dalam**

Proses pelatihan untuk jaringan neural dalam terdiri dari beberapa perulangan yang disebut epoch. Untuk epoch pertama, Anda mulai dengan menetapkan nilai inisialisasi acak untuk nilai bobot (w) dan bias b. Proses selanjutnya adalah sebagai berikut:

1. Fitur untuk pengamatan data dengan nilai label yang diketahui dikirimkan ke lapisan input. Umumnya, pengamatan tersebut dikelompokkan menjadi batch (sering kali disebut batch mini).
2. Setelah itu, neuron menerapkan fungsinya dan jika diaktifkan, teruskan hasilnya ke lapisan berikutnya sampai lapisan output menghasilkan prediksi.
3. Prediksi tersebut dibandingkan dengan nilai aktual yang diketahui dan jumlah varian antara nilai yang diprediksi dan nilai sebenarnya (yang kami sebut kerugian) dihitung.
4. Berdasarkan hasilnya, nilai yang direvisi untuk nilai bobot dan bias dihitung untuk mengurangi kerugian dan penyesuaian ini dipropagasi mundur ke neuron di dalam lapisan jaringan.
5. Epoch selanjutnya mengulangi propagasi maju pelatihan batch dengan nilai bobot dan bias yang direvisi dengan harapan meningkatkan akurasi model (dengan mengurangi kerugian).



1. **Memahami lebih lanjut fungsi kerugian dan propagasi mundur**

Deskripsi sebelumnya dari proses pelatihan pembelajaran dalam menyebutkan bahwa kerugian dari model dihitung dan digunakan untuk menyesuaikan nilai bobot dan bias. Bagaimana tepatnya hal ini bekerja?

* **Menghitung kerugian**

Misalkan salah satu sampel yang berhasil melalui proses pelatihan berisi fitur spesimen Adelie (kelas 0). Output yang benar dari jaringan adalah [1, 0, 0]. Sekarang misalkan output yang dihasilkan oleh jaringan adalah [0,4; 0,3; 0,3]. Dengan membandingkan output, kita dapat menghitung varian absolut untuk setiap elemen (dengan kata lain, seberapa jauh selisih setiap nilai yang diprediksi nilai yang seharusnya) seperti [0,6; 0,3; 0,3].

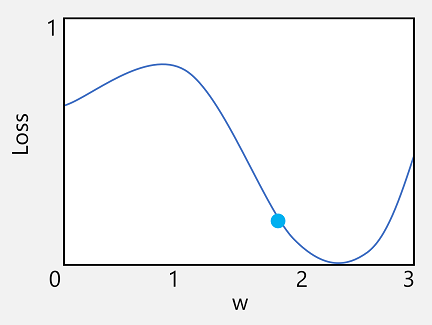
Pada kenyataannya, karena kita sebenarnya berurusan dengan beberapa pengamatan, kita biasanya mengagregasi varian - misalnya dengan menguadratkan nilai varian individu dan menghitung rata-ratanya, maka kita akan memperoleh satu nilai kerugian rata-rata, misalnya 0,18.

* **Pengoptimal**

Sekarang, inilah bagian yang cerdas. Kerugian dihitung menggunakan fungsi, yang beroperasi pada hasil dari lapisan akhir jaringan yang juga merupakan fungsi. Lapisan akhir jaringan beroperasi pada output dari lapisan sebelumnya, yang juga merupakan fungsi. Jadi, saat diterapkan, seluruh model mulai dari lapisan input hingga penghitungan kerugian hanyalah satu fungsi bertingkat besar. Fungsi memiliki beberapa karakteristik yang sangat berguna antara lain:

* Anda dapat membuat konsep fungsi sebagai garis yang dipetakan dan membandingkan outputnya dengan masing-masing variabel.
* Anda dapat menggunakan kalkulus diferensial untuk menghitung turunan dari fungsi pada titik tertentu sesuai variabelnya.

Mari membahas kemampuan yang pertama. Kita dapat memetakan garis fungsi untuk menunjukkan perbandingan nilai bobot individu dengan kerugian dan menandai pada garis di mana titik nilai bobot saat ini cocok dengan nilai kerugian saat ini.



Sekarang mari kita terapkan karakteristik kedua dari sebuah fungsi. Turunan dari fungsi untuk titik tertentu menunjukkan apakah kemiringan (atau gradien) dari output fungsi (dalam hal ini, kerugian) meningkat atau menurun terhadap variabel fungsi (dalam hal ini, nilai bobot). Turunan positif menunjukkan bahwa fungsi meningkat dan turunan negatif menunjukkan bahwa fungsi menurun. Dalam hal ini, pada titik yang dipetakan untuk nilai bobot saat ini, fungsi memiliki gradien menurun. Dengan kata lain, meningkatkan bobot akan menurunkan kerugian.

Kami menggunakan pengoptimal untuk menerapkan trik yang sama ke semua variabel bobot dan bias dalam model dan menentukan ke arah mana kita perlu menyesuaikannya (naik atau turun) untuk mengurangi jumlah kerugian keseluruhan dalam model. Ada beberapa algoritma pengoptimalan yang umum digunakan, termasuk stochastic gradient descent (SGD), Adaptive Learning Rate (ADADELTA), Adaptive Momentum Estimation (Adam), dan lain-lain; semuanya dirancang untuk mencari cara menyesuaikan bobot dan bias untuk memperkecil kerugian.

* **Tingkat pembelajaran**

Sekarang, pertanyaan paling relevan berikutnya adalah seberapa banyak pengoptimal harus menyesuaikan nilai bobot dan bias? Jika Anda melihat pemetaan untuk nilai bobot kita, tampak bahwa meningkatkan bobot dalam jumlah kecil akan diikuti oleh garis fungsi menurun (mengurangi kerugian). Namun, jika kita meningkatkannya terlalu banyak, garis fungsi mulai naik lagi, jadi mungkin kita sebenarnya akan meningkatkan kerugian; dan setelah epoch berikutnya, kita akan mendapati bahwa kita perlu mengurangi bobot.

Ukuran penyesuaian dikendalikan oleh parameter yang Anda tetapkan untuk pelatihan yang disebut tingkat pembelajaran. Tingkat pembelajaran yang rendah menghasilkan penyesuaian yang kecil (sehingga ia dapat melakukan lebih banyak epoch untuk memperkecil kerugian), sedangkan tingkat pembelajaran tinggi menghasilkan penyesuaian yang besar (sehingga Anda mungkin melewatkan semua minimum).

1. **Latihan - Melatih jaringan neural dalam**

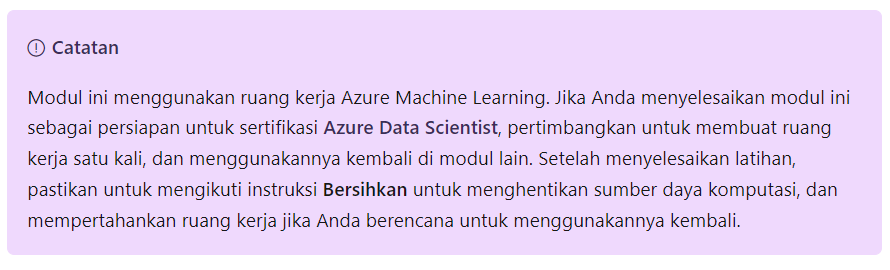
Dalam modul ini, Anda telah belajar banyak tentang teori dan prinsip pembelajaran dalam dengan jaringan neural. Cara terbaik untuk mempelajari cara menerapkan teori ini adalah dengan membangun model pembelajaran dalam yang sebenarnya dan itulah yang akan Anda lakukan dalam latihan ini.

Ada banyak kerangka kerja yang tersedia untuk melatih jaringan neural dalam dan dalam latihan ini Anda dapat memilih untuk menjelajahi salah satu (atau keduanya) dari dua kerangka pembelajaran dalam yang paling populer untuk Python: PyTorch dan TensorFlow.

1. **Sebelum memulai**

Untuk menyelesaikan latihan, Anda memerlukan:

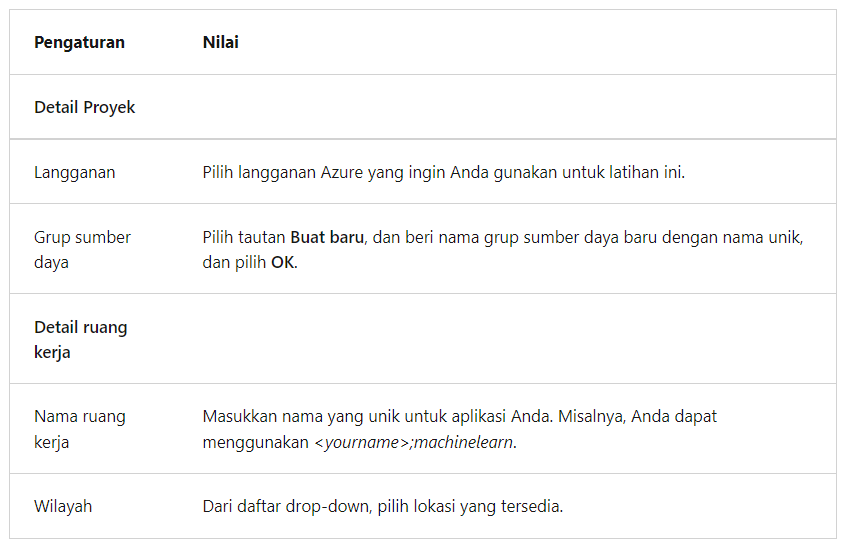
* Langganan Microsoft Azure. Jika belum memilikinya, Anda dapat mendaftar untuk uji coba gratis di https://azure.microsoft.com/free.
* Ruang kerja Azure Machine Learning dengan instans komputasi dan repositori ml-basics yang dikloning.



1. **Membuat ruang kerja Azure Machine Learning**

Jika Anda belum memiliki ruang kerja Azure Machine Learning di langganan Azure, ikuti langkah-langkah berikut untuk membuatnya:

1. Masuk ke portal Azure : <https://portal.azure.com/#home> menggunakan akun Microsoft yang terkait dengan langganan Azure Anda.
2. Di halaman Beranda Azure, pada bagian Layanan Azure, pilih Buat sumber daya. Panel Buat sumber daya akan muncul.
3. Di kotak pencarian Layanan pencarian dan marketplace, cari dan pilih Pembelajaran Mesin. Panel Pembelajaran Mesin muncul.
4. Pilih Buat. Panel Pembelajaran mesin muncul.
5. Pada tab Dasar, Masukkan nilai berikut ke setiap pengaturan.



1. Terima default yang tersisa, dan pilih Tinjau + buat.
2. Setelah melalui validasi, pilih Buat.
3. Tunggu hingga sumber daya ruang kerja Anda dibuat karena dapat memakan waktu beberapa menit.
4. Saat penyebaran selesai, pilih Buka sumber daya. Panel Pembelajaran mesin Anda muncul.
5. Pilih Luncurkan studio, atau buka https://ml.azure.com, dan masuk menggunakan akun Microsoft Anda. Halaman Microsoft Azure Machine Learning Studio muncul.
6. Di Azure Machine Learning Studio, hidupkan/matikan ☰ di kiri atas untuk meluaskan/menciutkan panel menunya. Anda dapat menggunakan opsi ini untuk mengelola sumber daya di ruang kerja Anda.
7. **Buat instans komputasi**

Untuk menjalankan catatan buku yang digunakan dalam latihan ini, Anda memerlukan instans komputasi di ruang kerja Azure Machine Learning.

* 1. Di panel menu sebelah kiri, pada bagian Kelola, pilih Komputasi. Panel Komputasi muncul.
  2. Pada tab Instans Komputasi, jika Anda sudah memiliki instans komputasi, mulailah; jika tidak, buat instans komputasi baru dengan memilih Baru. Panel Buat instans komputasi muncul.
  3. Masukkan nilai berikut untuk setiap pengaturan:

Nama komputasi: masukkan nama unik

Jenis mesin virtual: CPU

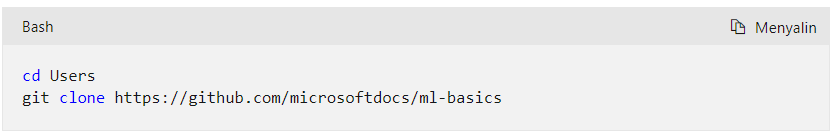
Ukuran mesin virtual: Pilih dari opsi yang disarankan: Standard\_DS11\_v2

* 1. Pilih Buat. Panel Komputasi muncul kembali dengan Instans komputasi Anda tercantum.
  2. Tunggu hingga instans komputasi dimulai karena ini mungkin memerlukan waktu beberapa menit. Pada bagian kolom Status, Instans komputasi Anda akan berubah menjadi Berjalan.

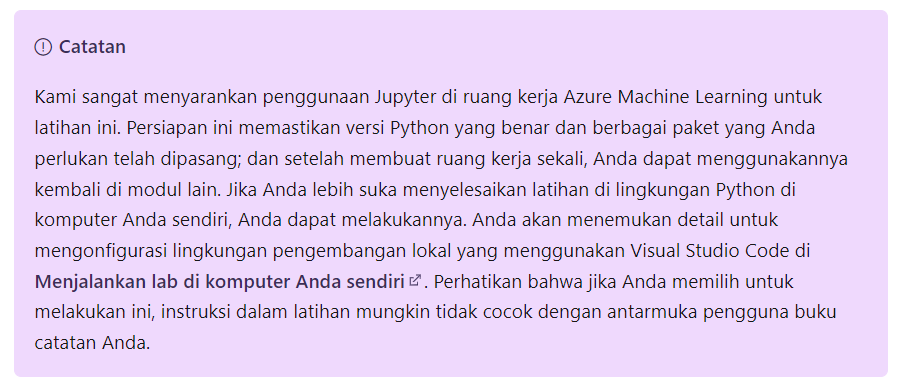
1. **Mengkloning repositori ml-basics**

File yang digunakan dalam modul ini, dan modul terkait lainnya, diterbitkan di repositori GitHub MicrosoftDocs/ml-basics. Jika Anda belum melakukannya, gunakan langkah-langkah berikut untuk mengkloning repositori ke ruang kerja Azure Machine Learning:

* 1. Pada bagian kolom Aplikasi, pilih tautan Jupyter untuk membuka Jupyter Notebooks di tab browser baru. Panel jupyter muncul.
  2. Pada bilah menu di paling kanan, pilih Baru, dan dari daftar dropdown, pilih Terminal. Sebuah shell terminal muncul.
  3. Jalankan perintah berikut untuk mengubah direktori saat ini menjadi direktori Pengguna, dan mengkloning repositori ml-basics, yang berisi buku catatan dan file yang akan Anda gunakan dalam latihan ini.



* 1. Setelah perintah selesai dan pemeriksaan file selesai, tutup tab terminal dan lihat beranda di penjelajah file catatan buku Jupyter Anda.
  2. Buka folder Pengguna - folder tersebut harus berisi folder ml-basics, yang berisi file yang akan Anda gunakan dalam modul ini.



1. **Melatih model jaringan neural dalam**

Setelah Anda membuat lingkungan Jupyter, dan mengkloning repositori ml-basics, Anda siap untuk menjelajahi pembelajaran mendalam.

1. Di Jupyter, dalam folder **ml-basics**, buka salah satu buku catatan **Deep Neural Networks (PyTorch).ipynb** atau **Deep Neural Networks (Tensorflow).ipynb** bergantung pada preferensi kerangka kerja Anda, dan ikuti petunjuk yang ada di dalamnya.
2. Setelah selesai, tutup dan hentikan semua catatan buku.

Setelah selesai mengerjakan buku catatan, kembali ke modul ini dan lanjutkan ke unit berikutnya untuk mempelajari lebih lanjut.

1. **Jaringan neural konvolusional**

Meskipun Anda dapat menggunakan model pembelajaran dalam untuk segala jenis pembelajaran mesin, sebenarnya model ini sangat berguna untuk menangani data yang terdiri dari larik besar dari nilai numerik - seperti gambar. Model pembelajaran mesin yang bekerja dengan gambar adalah fondasi untuk area kecerdasan buatan yang disebut computer vision dan teknik pembelajaran dalam telah memainkan peran utama untuk mendorong kemajuan luar biasa di bidang ini selama beberapa tahun terakhir.

Yang menjadi inti kesuksesan pembelajaran dalam di bidang ini adalah sejenis **model** yang disebut jaringan neural konvolusional atau **CNN**. CNN biasanya bekerja dengan mengekstrak fitur dari gambar kemudian memasukkan fitur-fitur itu ke jaringan neural yang terhubung sepenuhnya untuk menghasilkan prediksi. Lapisan ekstraksi fitur dalam jaringan memiliki efek mengurangi jumlah fitur dari larik nilai piksel individu yang berpotensi berjumlah besar ke set fitur yang lebih kecil yang mendukung prediksi label.

1. **Lapisan dalam CNN**

CNN terdiri dari beberapa lapisan, masing-masing melakukan tugas tertentu dalam mengekstrak fitur atau memprediksi label.

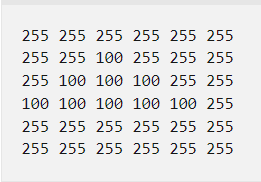
1. **Lapisan konvolusional**

Salah satu jenis lapisan utama adalah lapisan konvolusional yang mengekstrak fitur penting dalam gambar. Lapisan konvolusional bekerja dengan menerapkan filter ke gambar. Filter ditentukan oleh kernel yang terdiri dari matriks nilai bobot.

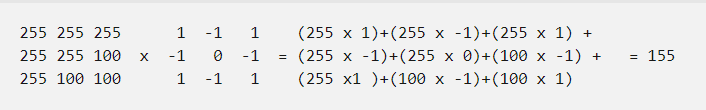
Misalnya, filter 3x3 mungkin ditentukan seperti ini:



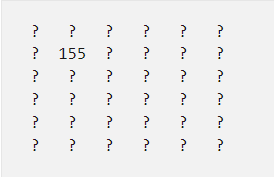
Gambar juga hanyalah sebuah matriks dari nilai piksel. Untuk menerapkan filter, Anda "menempatkan" filter ke gambar dan menghitung jumlah tertimbang dari nilai piksel gambar yang sesuai di bagian kernel filter. Gambar hasil kemudian ditetapkan ke sel tengah patch 3x3 yang setara ke dalam matriks nilai baru yang berukuran sama dengan gambar. Misalnya, gambar 6 x 6 memiliki nilai piksel berikut:



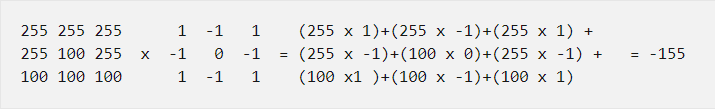
Menerapkan filter ke patch 3x3 sebelah kiri atas gambar akan terlihat seperti ini:



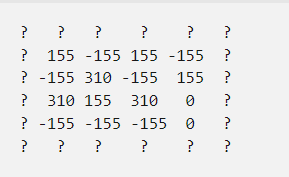
Hasil tersebut ditetapkan ke nilai piksel terkait dalam matriks baru seperti ini:



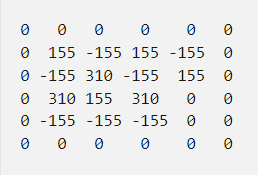
Sekarang filter dipindahkan secara bersamaan (dikonvolusi), biasanya menggunakan ukuran langkah 1 (sehingga bergerak sepanjang satu piksel ke kanan), dan nilai untuk piksel berikutnya akan dihitung



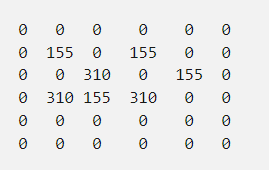
Jadi, sekarang kita bisa mengisi nilai berikutnya dari matriks yang baru.



Karena ukuran kernel filter, kita tidak dapat menghitung nilai untuk piksel di bagian tepi; jadi, biasanya kita hanya menerapkan nilai padding (sering kali 0):



Output konvolusi biasanya diteruskan ke fungsi aktivasi yang seringkali merupakan fungsi Rectified Linear Unit (ReLU) yang memastikan nilai negatif diatur ke 0:



Matriks yang dihasilkan adalah peta fitur dari nilai fitur yang dapat digunakan untuk melatih model pembelajaran mesin.

**Catatan**: Nilai dalam peta fitur bisa lebih besar dari nilai maksimum untuk piksel (255). Jadi, jika Anda ingin memvisualisasikan peta fitur sebagai gambar, Anda harus menormalkan nilai fitur antara 0 dan 255.

**Proses konvolusi ditampilkan dalam animasi di bawah ini.**

**File Gif** : Data Scientist Indonesia - Cloud Skills Challenge from Microsoft\5 - Melatih dan mengevaluasi model pembelajaran mendalam\convolution

**Penjelasan :**

* 1. Gambar diteruskan ke lapisan konvolusional. Dalam hal ini, gambar adalah bentuk geometris sederhana.
  2. Gambar terdiri dari larik piksel dengan nilai antara 0 dan 255 (untuk gambar berwarna, biasanya berbentuk larik 3 dimensi dengan nilai saluran merah, hijau, dan biru).
  3. Kernel filter umumnya diinisialisasi dengan bobot acak (dalam contoh ini, kami telah memilih nilai untuk menyoroti efek yang mungkin dihasilkan filter pada nilai piksel; tetapi dalam CNN yang nyata, bobot awal biasanya akan dihasilkan dari distribusi Gaussian acak). Filter ini akan digunakan untuk mengekstrak peta fitur dari data gambar.
  4. Filter dikonvolusi ke seluruh gambar, sehingga menghitung nilai fitur dengan menerapkan jumlah bobot dikalikan dengan nilai piksel terkait di setiap posisi. Fungsi aktivasi Rectified Linear Unit (ReLU) diterapkan untuk memastikan nilai negatif diatur ke 0.
  5. Setelah konvolusi, peta fitur berisi nilai fitur yang diekstraksi yang sering kali menekankan atribut visual utama dari gambar. Dalam hal ini, peta fitur menyoroti bagian tepi dan sudut segitiga dalam gambar.

Biasanya, lapisan konvolusional menerapkan beberapa kernel filter. Setiap filter menghasilkan peta fitur yang berbeda dan semua peta fitur diteruskan ke lapisan jaringan berikutnya.

1. **Lapisan penggabungan**

Setelah mengekstrak nilai fitur dari gambar, lapisan penggabungan (atau pengurangan sampel) digunakan untuk mengurangi jumlah nilai fitur sambil tetap mempertahankan fitur pembeda utama yang telah diekstrak.

Salah satu jenis penggabungan yang paling umum adalah penggabungan maks di mana filter diterapkan ke gambar dan hanya nilai piksel maksimum dalam area filter yang dipertahankan. Jadi misalnya, menerapkan kernel penggabungan 2x2 ke patch gambar berikut akan menghasilkan 155.



Perhatikan bahwa efek filter penggabungan 2x2 adalah mengurangi jumlah nilai dari 4 ke 1.

Seperti halnya lapisan konvolusional, lapisan penggabungan bekerja dengan menerapkan filter ke seluruh peta fitur. Animasi di bawah ini menunjukkan contoh penggabungan maks untuk peta gambar.

**File Gif :**  In folder name Pooling

1. Peta fitur yang diekstrak oleh filter dalam lapisan konvolusional berisi larik nilai fitur.
2. Kernel penggabungan digunakan untuk mengurangi jumlah nilai fitur. Dalam hal ini, ukuran kernel adalah 2x2 sehingga akan menghasilkan larik dengan seperempat dari jumlah nilai fitur.
3. Kernel penggabungan dikonvolusi di seluruh peta fitur, hanya mempertahankan nilai piksel tertinggi di setiap posisi.
4. **Lapisan penghapusan**

Salah satu tantangan paling sulit dalam CNN adalah menghindari overfitting, di mana model yang dihasilkan berperforma baik dengan data pelatihan tetapi tidak dapat digeneralisasikan dengan baik untuk data baru yang tidak digunakan dalam pelatihan. Salah satu teknik yang dapat Anda gunakan untuk mengurangi overfitting adalah memasukkan lapisan di mana proses pelatihan secara acak menghilangkan (atau "menghapus") peta fitur. Hal tersebut mungkin tampak tidak masuk akal, tetapi inilah cara yang efektif untuk memastikan bahwa model tidak menjadi terlalu bergantung pada gambar pelatihan.

Teknik lain yang dapat Anda gunakan untuk mengurangi overfitting adalah membalik, mencerminkan, atau memiringkan gambar pelatihan secara acak untuk menghasilkan data yang bervariasi di antara epoch pelatihan.

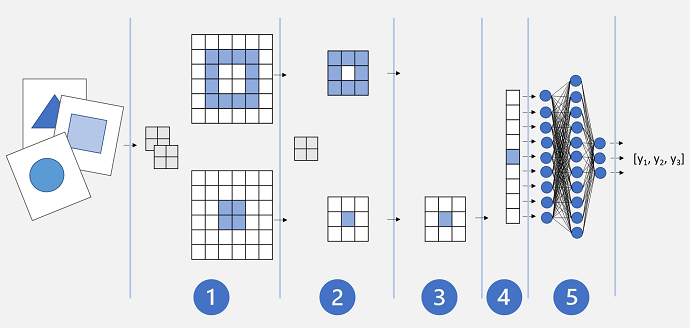
1. **Lapisan perataan**

Setelah menggunakan lapisan konvolusional dan penggabungan untuk mengekstrak fitur penting dalam gambar, peta fitur yang dihasilkan adalah larik nilai piksel multidimensi. Lapisan perataan digunakan untuk meratakan peta fitur menjadi vektor nilai yang dapat digunakan sebagai input ke lapisan yang terhubung sepenuhnya.

1. **Lapisan yang terhubung sepenuhnya**

Biasanya, CNN diakhiri dengan jaringan yang terhubung sepenuhnya di mana nilai fitur diteruskan ke lapisan input melalui satu atau beberapa lapisan tersembunyi dan menghasilkan nilai yang diprediksi dalam lapisan output.

Arsitektur CNN dasar mungkin terlihat mirip dengan ini:



1. Gambar diberikan ke lapisan konvolusional. Dalam hal ini, ada dua filter, sehingga setiap gambar menghasilkan dua peta fitur.
2. Peta fitur diteruskan ke lapisan penggabungan, di mana kernel penggabungan 2x2 mengurangi ukuran peta fitur.
3. Lapisan penghapusan secara acak menghapus beberapa peta fitur untuk mencegah overfitting.
4. Lapisan perataan mengambil larik peta fitur yang tersisa dan meratakannya menjadi vektor.
5. Elemen vektor diumpankan ke jaringan yang terhubung sepenuhnya, yang menghasilkan prediksi. Dalam hal ini, jaringan adalah model klasifikasi yang memprediksi peluang untuk tiga kelas gambar yang mungkin (segitiga, persegi, dan lingkaran).

**Melatih model CNN**

Seperti halnya jaringan neural dalam, CNN dilatih dengan meneruskan batch data pelatihan melalui beberapa epoch, menyesuaikan nilai bobot dan nilai bias berdasarkan kerugian yang dihitung untuk setiap epoch. Dalam kasus CNN, propagasi mundur dari bobot yang disesuaikan mencakup bobot kernel filter yang digunakan dalam lapisan konvolusional serta bobot yang digunakan dalam lapisan yang terhubung sepenuhnya.

**1. Latihan - Melatih jaringan neural konvolusional**

PyTorch dan TensorFlow menawarkan dukungan komprehensif untuk membangun jaringan neural konvolusional sebagai model klasifikasi untuk gambar.

Dalam latihan ini, Anda akan menggunakan kerangka kerja pilihan Anda untuk membuat pengklasifikasi gambar berbasis CNN sederhana untuk gambar berbentuk geometris sederhana. Prinsip yang sama dapat diterapkan pada gambar dalam bentuk apa pun.

Untuk menyelesaikan latihan:

1. Di Jupyter, di folder **ml-basics**, buka notebook **Jaringan Neural Konvolusional (PyTorch).ipynb** atau **Jaringan Neural Konvolusional (Tensorflow).ipynb**, bergantung pada preferensi kerangka kerja Anda, dan ikuti instruksi di dalamnya.
2. Setelah selesai, tutup dan hentikan semua buku catatan.

Setelah selesai mengerjakan buku catatan, kembali ke modul ini dan lanjutkan ke unit berikutnya untuk mempelajari lebih lanjut.

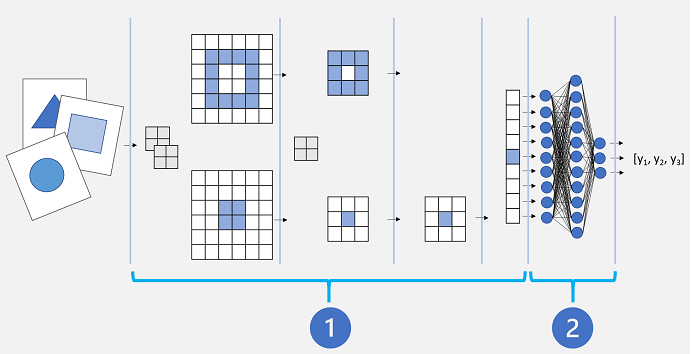
**E. Pembelajaran transfer**

Dalam hidup, sering kali lebih mudah untuk mempelajari keterampilan baru jika Anda sudah memiliki keahlian dalam keterampilan yang serupa dan dapat ditransfer. Misalnya, mungkin lebih mudah untuk mengajari seseorang cara mengendarai bus jika dia sudah mempelajari cara mengendarai mobil. Pengemudi dapat menggunakan keterampilan mengemudi mobil yang telah dipelajari dan menerapkannya untuk mengemudikan bus.

Prinsip yang sama dapat diterapkan pada pelatihan model pembelajaran dalam melalui teknik yang disebut pemelajaran transfer.

1. **Cara kerja pemelajaran transfer**

Jaringan Neural Konvolusional (CNN) untuk klasifikasi gambar biasanya terdiri dari beberapa lapisan yang mengekstrak fitur lalu menggunakan lapisan akhir yang terhubung sepenuhnya untuk mengklasifikasikan gambar berdasarkan fitur-fitur ini.



Secara konseptual, jaringan neural terdiri dari dua set lapisan yang berbeda:

1. Satu set lapisan dari model dasar yang melakukan ekstraksi fitur.
2. Lapisan yang terhubung sepenuhnya yang mengambil fitur yang diekstraksi dan menggunakannya untuk prediksi kelas.

Lapisan ekstraksi fitur menerapkan filter konvolusional dan penggabungan untuk menekankan tepi, sudut, dan pola lain dalam gambar yang dapat digunakan untuk membedakannya, dan secara teori harus berfungsi untuk setiap set gambar dengan dimensi yang sama dengan lapisan input jaringan. Lapisan prediksi memetakan fitur ke serangkaian output yang mewakili probabilitas setiap label kelas yang ingin Anda gunakan untuk mengklasifikasikan gambar.

Dengan memisahkan jaringan ke dalam jenis lapisan ini, kita dapat mengambil lapisan ekstraksi fitur dari model yang telah dilatih dan menambahkan satu lapisan atau lebih guna memakai fitur yang diekstraksi untuk prediksi label kelas yang sesuai bagi gambar Anda. Dengan pendekatan ini, Anda dapat menjaga bobot terlatih untuk lapisan ekstraksi fitur. Artinya, Anda hanya perlu melatih lapisan prediksi yang telah Anda tambahkan.

Ada banyak arsitektur jaringan neural konvolusional yang andal untuk klasifikasi gambar yang dapat Anda gunakan sebagai model dasar untuk pemelajaran transfer, sehingga Anda dapat membangun pada pekerjaan yang dilakukan oleh orang lain agar dapat dengan mudah membuat model klasifikasi gambar yang efektif.

# **2. Latihan - Menggunakan pemelajaran transfer**

PyTorch dan TensorFlow mendukung pustaka model yang sudah ada yang dapat Anda gunakan sebagai dasar untuk pemelajaran transfer.

Dalam latihan ini, Anda akan menggunakan kerangka kerja pilihan Anda untuk melatih model jaringan neural konvolusional dengan menggunakan pemelajaran transfer.

**Untuk menyelesaikan latihan:**

1. Di Jupyter, dalam folder **ml-basics**, buka baik buku catatan **Transfer Learning (PyTorch).ipynb** atau **Transfer Learning (Tensorflow).ipynb**, bergantung pada preferensi kerangka kerja Anda, dan ikuti instruksi yang ada di dalamnya.
2. Setelah selesai, tutup dan hentikan semua catatan buku.

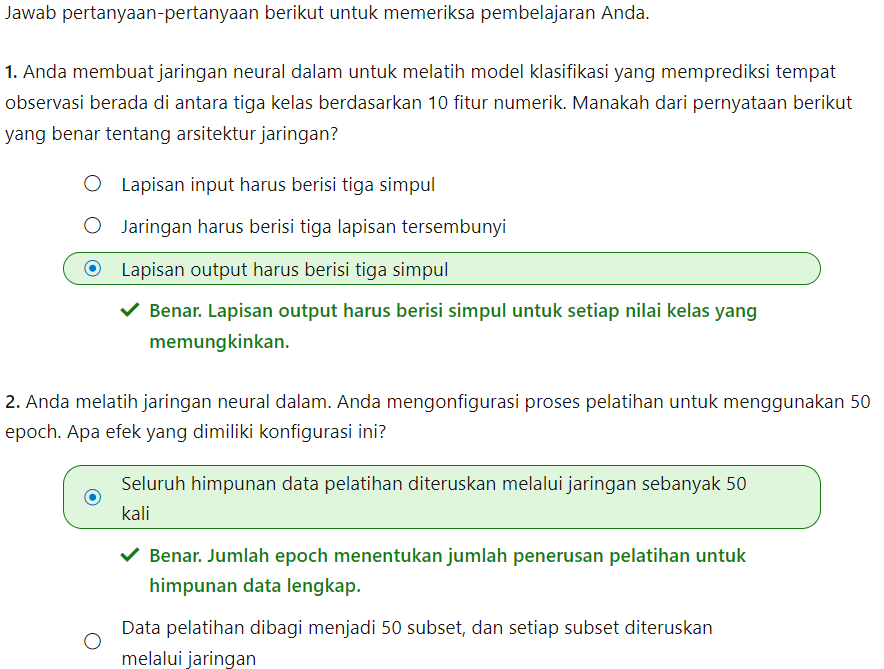
**Pembersihan**

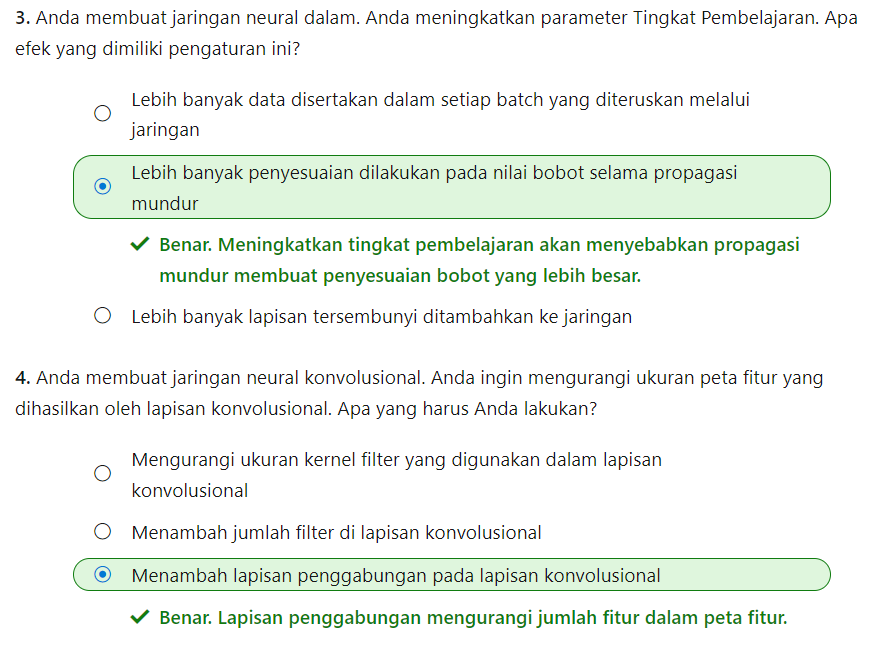
Jika Anda **menggunakan instans komputasi** di ruang kerja Azure Machine Learning untuk menyelesaikan latihan, gunakan langkah-langkah ini untuk membersihkan.

1. Tutup semua catatan buku Jupyter dan beranda Jupyter.
2. Di Studio Azure Machine Learning, pada halaman Komputasi, pilih instans komputasi Anda, dan di bilah menu atas, pilih Hentikan.

Jika Anda tidak bermaksud untuk menyelesaikan modul lain yang memerlukan ruang kerja Azure Machine Learning, Anda dapat menghapus grup sumber daya yang Anda buat untuknya dari langganan Azure.

**UJI Pengetahuan :**





**Ringkasan**

Dalam modul ini, Anda telah mempelajari prinsip dasar pembelajaran dalam dan cara membuat model jaringan neural dalam menggunakan PyTorch atau Tensorflow. Anda juga telah mempelajari penggunaan jaringan neural konvolusional untuk membuat model klasifikasi gambar.

Teknik pembelajaran dalam menjadi ujung tombak pembelajaran mesin dan kecerdasan buatan, serta digunakan untuk mengimplementasikan solusi perusahaan. Jika modul ini telah menginspirasi Anda untuk membangun solusi pembelajaran mesin, pertimbangkan untuk mempelajari bagaimana Azure Machine Learning dapat membantu Anda melatih, menyebarkan, dan mengelola model dalam skala besar. Anda dapat mempelajari cara menggunakan Azure Machine Learning untuk mengelola operasi pembelajaran mesin di jalur pembelajaran Membangun solusi AI dengan Azure Machine Learning service.

<https://learn.microsoft.com/id-id/training/paths/build-ai-solutions-with-azure-ml-service/>