Materi untuk landasan teori

* Sentiment Analysis
* Jaringan syaraf tiruan
* Deep Learning
  + Optimizer
  + Activation function
  + Regularization
  + Loss function
* Word Embedding
* LSTM
* BILSTM

Pada Bab II, akan dijelaskan landasan teori terkait penelitian ini, seperti Sentiment Analysis, Deep Learning, Word Embedding, LSTM dan BiLSTM.

1. Sentiment Analysis

Sentiment analysis merupakan penggunaan *natural language processing*, analisis teks, linguistik komputasi, dan biometrik untuk mengidentifikasi, mengekstrak, mengukur, dan mempelajari keadaan afektif dan informasi subjektif secara sistematis. Dengan kata yang lain, sentiment analysis digunakan untuk menentukan status suatu data, yang bisa mengandung muatan positif, netral ataupun negatif.

Sumber:

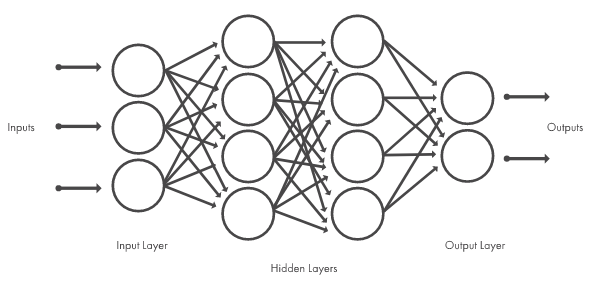
[1] <https://algorit.ma/blog/sentiment-analysis-adalah-2022/>

[2] <https://en.wikipedia.org/wiki/Sentiment_analysis>

[3] <https://www.sciencedirect.com/science/article/pii/S2090447914000550>

1. Deep Learning

*Deep learning* merupakan teknik pembelajaran mesin yang mengajarkan komputer untuk melakukan apa yang terjadi secara alami pada manusia, atau bisa dikatakan belajar melalui sebuah contoh. Dalam *deep learning*, model komputer belajar untuk melakukan tugas klasifikasi langsung dari gambar, teks, atau suara. Model pembelajaran mendalam dapat mencapai akurasi mutakhir, terkadang melebihi kinerja tingkat manusia. Model dilatih dengan menggunakan sekumpulan besar data berlabel dan arsitektur jaringan saraf yang berisi banyak lapisan. Hampir semua metode *deep learning* menggunakan arsitektur jaringan syaraf tiruan. Istilah "deep" biasanya mengacu pada jumlah lapisan tersembunyi di jaringan saraf. Jaringan saraf tradisional hanya berisi 2-3 lapisan tersembunyi, sedangkan jaringan dalam dapat memiliki sebanyak 150. Model *deep learning* dilatih menggunakan dataset berlabel dalam jumlah yang banyak dan arsitektur jaringan syaraf yang belajar langsung dari data tanpa dilakukannya sebuah ekstraksi fitur.



[1] [https://www.mathworks.com/discovery/deep-learning.html#:~:text=Deep%20learning%20is%20a%20machine,a%20pedestrian%20from%20a%20lam ppost](https://www.mathworks.com/discovery/deep-learning.html#:~:text=Deep%20learning%20is%20a%20machine,a%20pedestrian%20from%20a%20lamppost).

1. Optimizer

[1] <https://medium.com/mlearning-ai/optimizers-in-deep-learning-7bf81fed78a0>

[2] <https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/#:~:text=While%20training%20the%20deep%20learning,loss%20and%20improve%20the%20accuracy>.

[3] <https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6>

Optimizer adalah algoritma atau metode yang digunakan untuk mengubah atribut jaringan saraf Anda seperti bobot *learning rate* untuk mengurangi loss. Algoritma atau strategi optimizer bertanggung jawab untuk mengurangi loss dan memberikan hasil yang seakurat mungkin. Berikut beberapa tipe optimizer yang sering digunakan

1. RMS prop

RMS prop adalah salah satu optimizer populer di kalangan *deep learning*. RMSprop menggunakan tanda gradien yang mengadaptasi ukuran langkah secara individual untuk setiap bobot. Dalam algoritma ini, dua gradien pertama dibandingkan untuk tanda-tanda. Jika mereka memiliki tanda yang sama, kita menuju ke arah yang benar dan karenanya meningkatkan ukuran langkah dengan sebagian kecil. Sedangkan jika memiliki tanda yang berlawanan, kita harus mengurangi ukuran langkah. Kemudian kami membatasi ukuran langkah, dan sekarang kami dapat melakukan pembaruan berat.

Masalah dengan RPROP adalah tidak berfungsi dengan baik dengan kumpulan data besar dan ketika ingin melakukan pembaruan mini-batch. Jadi, pencapaian kekokohan RPPROP dan efisiensi mini-batch pada saat yang sama adalah motivasi utama di balik munculnya RMS prop. RMS prop juga dapat dianggap sebagai kemajuan dalam pengoptimal AdaGrad karena mengurangi tingkat pembelajaran yang menurun secara monoton.

Algoritma ini terutama berfokus pada percepatan proses optimasi dengan mengurangi jumlah evaluasi fungsi untuk mencapai minimum lokal. Algoritma menjaga rata-rata bergerak gradien kuadrat untuk setiap bobot dan membagi gradien dengan akar kuadrat dari kuadrat rata-rata.

RMS Prop(Root Mean Square) Deep Learning Optimizer

di mana gamma adalah faktor pelupa. Bobot diperbarui dengan rumus di bawah ini

RMS Prop(Root Mean Square) Deep Learning Optimizer

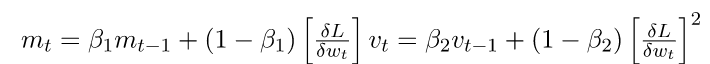
Dalam istilah yang lebih sederhana, jika ada parameter yang menyebabkan fungsi cost banyak berosilasi, kami ingin menghukum (penalize) pembaruan parameter ini. Misalkan Anda membuat model untuk mengklasifikasikan berbagai jenis ikan. Model bergantung pada faktor 'warna' terutama untuk membedakan antara ikan. Karena itu membuat banyak kesalahan. Apa yang dilakukan RMS Prop adalah, menghukum (penalize) parameter 'warna' sehingga dapat mengandalkan fitur lain juga. Ini mencegah algoritma beradaptasi terlalu cepat terhadap perubahan parameter 'warna' dibandingkan dengan parameter lainnya. Algoritma ini memiliki beberapa keunggulan dibandingkan dengan versi sebelumnya dari algoritma penurunan gradien. Algoritma konvergen dengan cepat dan membutuhkan penyetelan yang lebih sedikit daripada algoritme penurunan gradien dan variannya.

Masalah dengan RMS Prop adalah bahwa tingkat pembelajaran harus ditentukan secara manual dan nilai yang disarankan tidak berfungsi untuk setiap aplikasi.

1. Adam

Algoritma optimasi ini merupakan perluasan lebih lanjut dari stocasthic gradient descent untuk memperbarui bobot jaringan selama training. Tidak seperti mempertahankan tingkat pembelajaran tunggal melalui pelatihan dalam SGD, optimizer Adam memperbarui tingkat pembelajaran untuk setiap bobot jaringan secara individual. Dalam adam, alih-alih mengadaptasi tingkat pembelajaran berdasarkan momen pertama (rata-rata) seperti di RMS Prop, ia juga menggunakan momen kedua dari gradien.

Algoritma ini mudah diterapkan, memiliki waktu berjalan yang lebih cepat, kebutuhan memori yang rendah, dan memerlukan penyetelan yang lebih sedikit daripada algoritme pengoptimalan lainnya.



Rumus di atas merupakan cara kerja pengoptimal adam. Di sini B1 dan B2 mewakili laju peluruhan rata-rata gradien.

Kelemahan optimizer Ini cenderung fokus pada waktu komputasi yang lebih cepat, sedangkan algoritma seperti penurunan gradien stokastik fokus pada titik data. Itu sebabnya algoritme seperti SGD menggeneralisasi data dengan cara yang lebih baik dengan mengorbankan kecepatan komputasi yang rendah. Jadi, algoritme pengoptimalan dapat dipilih sesuai dengan kebutuhan dan jenis data.

1. Activation function

*Activation function* menentukan apakah neuron dalam sebuah jaringan syaraf *deep learning* harus diaktifkan ataupun tidak. Ini menyiratkan bahwa ia akan menggunakan beberapa operasi matematika sederhana untuk menentukan apakah input neuron ke jaringan relevan atau tidak relevan dalam proses prediksi. Kemampuan untuk memperkenalkan non-linearitas ke jaringan saraf tiruan dan menghasilkan output dari kumpulan nilai input yang diumpankan ke lapisan adalah tujuan dari fungsi aktivasi.

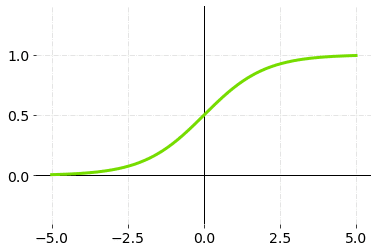
Berikut beberapa contoh *activation function*:

1. Sigmoid

*Activation function* sigmoid menerima masukan nilai dan melakukan *return*  0 ataupun 1. Sigmoid biasanya digunakan untuk menangani klasifikasi biner. Sigmoid memberikan probabilitas keberadaan kelas tertentu. Jika direpresentasikan dalam matematika, maka sigmoid akan menjadi persamaan seperti berikut:

Diagram, schematic

Description automatically generated



[1]<https://www.kdnuggets.com/2022/06/activation-functions-work-deep-learning.html>

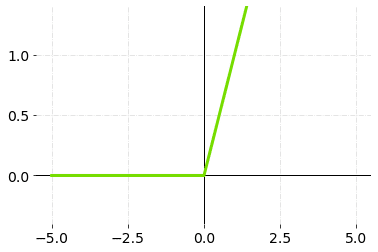
1. ReLu(Rectified Linear Unit)

ReLU adalah singkatan dari Rectified Linear Unit dan merupakan salah satu fungsi aktivasi yang paling umum digunakan dalam aplikasi. Re-Lu memecahkan masalah gradient yang hilang karena nilai maksimun gradien fungsi re-lu adalah 1. Ini juga memecahkan masalah saturasi neuron, karena kemiringan tidak pernah nol untuk fungsi ReLU. Kisaran ReLU adalah antara 0 dan tak terhingga.

Jika direpresentasikan dalam matematika, maka Re-Lu akan menjadi persamaan seperti berikut:

Text

Description automatically generated



<https://www.kdnuggets.com/2022/06/activation-functions-work-deep-learning.html>

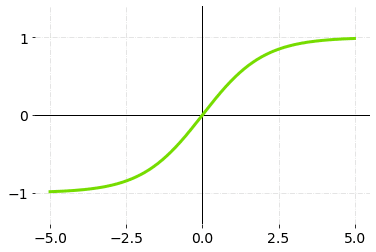
1. Softmax

Kombinasi dari banyak sigmoid disebut sebagai fungsi Softmax. Ini menentukan probabilitas relatif. Serupa dengan fungsi aktivasi sigmoid, fungsi Softmax mengembalikan probabilitas setiap kelas/label. Dalam klasifikasi multi-kelas, fungsi aktivasi softmax paling sering digunakan untuk lapisan terakhir dari jaringan saraf. Fungsi softmax memberikan probabilitas kelas saat ini sehubungan dengan yang lain. Ini berarti bahwa ia juga mempertimbangkan kemungkinan kelas lain juga.

Jika direpresentasikan dalam matematika, maka softmax akan menjadi persamaan seperti berikut:

Text

Description automatically generated



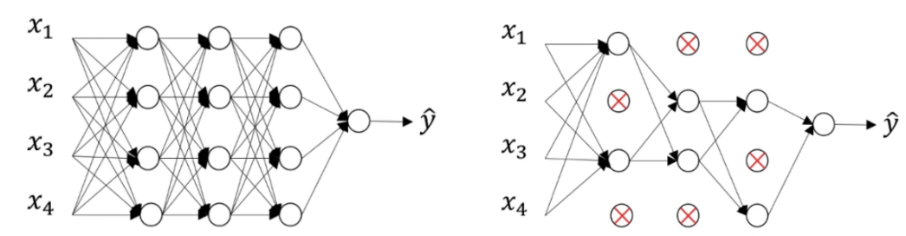
1. Regularization

[1] <https://towardsdatascience.com/regularization-in-deep-learning-l1-l2-and-dropout-377e75acc036>  
Regularisasi mengacu pada serangkaian teknik berbeda yang menurunkan kompleksitas model jaringan saraf selama pelatihan, dan dengan demikian mencegah overfitting.

Salah satu Teknik regularisasi yang populer adalah Dropout

1. Dropout

Singkatnya, dropout berarti bahwa selama pelatihan dengan beberapa probabilitas P neuron dari jaringan saraf dimatikan selama pelatihan. Mari kita lihat contoh visualnya.



Asumsikan di sisi kiri kita memiliki jaringan saraf feedforward tanpa dropout. Menggunakan dropout dengan katakanlah probabilitas P=0,5 bahwa neuron acak dimatikan selama pelatihan akan menghasilkan jaringan saraf di sisi kanan.

Dalam hal ini, dapat diamati bahwa sekitar setengah dari neuron tidak aktif dan tidak dianggap sebagai bagian dari jaringan saraf. Dan seperti yang telah diamati, jaringan saraf menjadi lebih sederhana.

Versi jaringan saraf yang lebih sederhana menghasilkan lebih sedikit kerumitan yang dapat mengurangi overfitting. Penonaktifan neuron dengan probabilitas tertentu P diterapkan pada setiap langkah perambatan maju dan pembaruan bobot.

1. Loss function

[1] <https://towardsdatascience.com/loss-functions-and-their-use-in-neural-networks-a470e703f1e9>

*Loss function* adalah fungsi yang membandingkan target dan nilai keluaran yang diprediksi; mengukur seberapa baik jaringan saraf memodelkan data pelatihan. Saat pelatihan, kami bertujuan untuk meminimalkan kerugian ini antara output yang diprediksi dan target. Dalam jaringan syaraf tiruan, terdapat 2 jenis *loss function, yaitu* regresi dan classification loss functions.

1. Regression Loss Functions

digunakan dalam jaringan saraf regresi, diberi nilai input, model memprediksi nilai output yang sesuai (bukan label yang dipilih sebelumnya. Sebagai contoh, terdapat Mean Squared Error (MSE) dan Mean Absolute Error (MAE)

1. MSE

Text

Description automatically generated  
Fungsi ini memiliki banyak properti yang membuatnya sangat cocok untuk menghitung loss. Selisihnya kuadrat, yang berarti tidak masalah apakah nilai prediksi di atas atau di bawah nilai target; namun, nilai dengan kesalahan besar akan dihukum. MSE juga merupakan fungsi cembung (seperti yang ditunjukkan pada diagram di atas) dengan minimum global yang ditentukan dengan jelas — ini memungkinkan kita untuk lebih mudah memanfaatkan optimasi penurunan gradien untuk mengatur nilai bobot.

Namun, satu kelemahan dari loss function ini adalah sangat sensitif terhadap outlier, yang berarti jika nilai prediksi secara signifikan lebih besar dari atau kurang dari nilai targetnya, ini akan meningkatkan kerugian secara signifikan.

1. MAE

Text

Description automatically generated with medium confidence

Loss function ini digunakan sebagai alternatif untuk MSE dalam beberapa kasus. Seperti disebutkan sebelumnya, MSE sangat sensitif terhadap outlier, yang secara dramatis dapat mempengaruhi kerugian karena jarak kuadrat. MAE digunakan dalam kasus ketika data pelatihan memiliki sejumlah besar outlier untuk mengurangi hal ini. Loss function ini juga memiliki beberapa kelemahan; karena jarak rata-rata mendekati 0, optimasi penurunan gradien tidak akan bekerja, karena turunan fungsi pada 0 tidak terdefinisi (yang akan menghasilkan kesalahan, karena tidak mungkin untuk membagi dengan 0).

1. Classification Loss Functions

Loss function classification digunakan dalam jaringan saraf klasifikasi dimana diberikan input, jaringan saraf menghasilkan vektor probabilitas input yang termasuk dalam berbagai kategori yang telah ditentukan sebelumnya kemudian dapat memilih kategori dengan probabilitas kepemilikan tertinggi; Sebagai contoh, loss function binary crossentropy dan categorical crossentropy

1. Binary crossentropy

Jaringan saraf klasifikasi bekerja dengan mengeluarkan vektor probabilitas dimana probabilitas bahwa input yang diberikan cocok dengan masing-masing kategori yang telah ditentukan sebelumnya; kemudian memilih kategori dengan probabilitas tertinggi sebagai hasil akhir.

Dalam klasifikasi biner, hanya ada dua kemungkinan nilai aktual y adalah 0 atau 1. Jadi, untuk secara akurat menentukan kerugian antara nilai aktual dan prediksi, perlu membandingkan nilai aktual (0 atau 1) dengan probabilitas bahwa input sejajar dengan kategori itu (p(i) = probabilitas bahwa kategorinya adalah 1; 1 — p(i) = probabilitas bahwa kategorinya adalah 0)

Text

Description automatically generated with low confidence

1. Categorical crossentropy

Dalam kasus di mana jumlah kelas lebih dari dua, akan menggunakan categorical crossentropy . Categorical crossentropy mengikuti proses yang sangat mirip dengan binary crossentropy, yang membedakan hanyalah jumlah kelas yang ada.

Diagram, text

Description automatically generated

1. LSTM

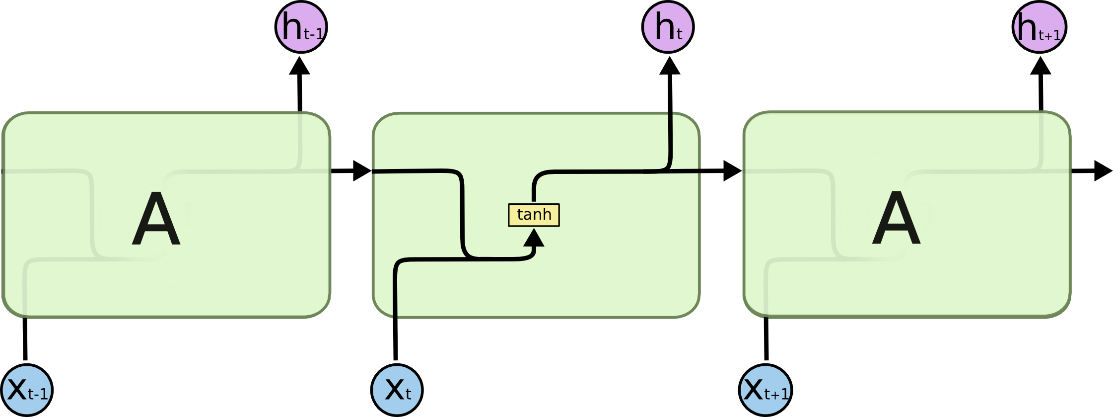
[1] <https://rifqifai.com/cara-kerja-long-short-term-memory-lstm/>

[2] <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

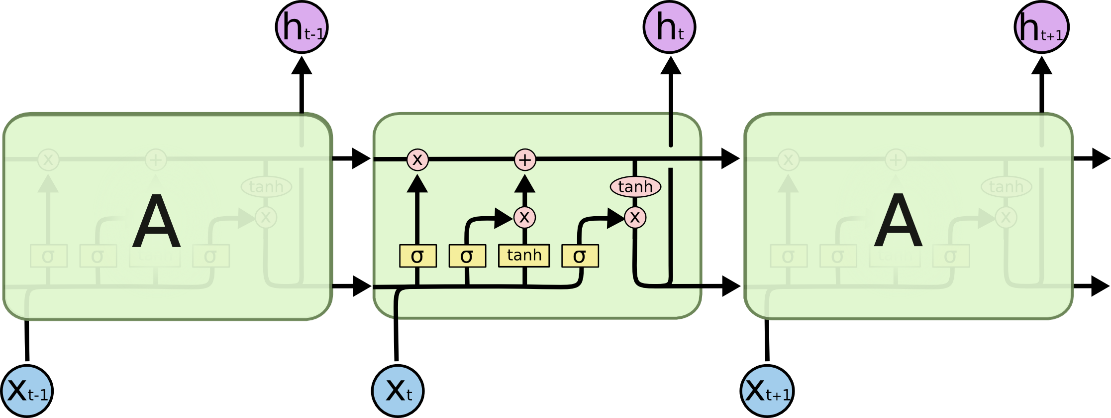
Jaringan Long Short Term Memory – biasanya hanya disebut “LSTMs” – adalah jenis khusus dari RNN, .

Jaringan Long Short Term Memory, atau biasanya disebut LSTM, merupakan pegembangan dari RNN (Recurent Neural Network) yang mampu mempelajari dependensi jangka panjang

LSTM secara eksplisit dirancang untuk menghindari masalah ketergantungan jangka panjang. Mengingat informasi untuk jangka waktu yang lama secara praktis merupakan perilaku standar mereka. Semua jenuis RNN memiliki bentuk rantai modul berulang jaringan saraf.

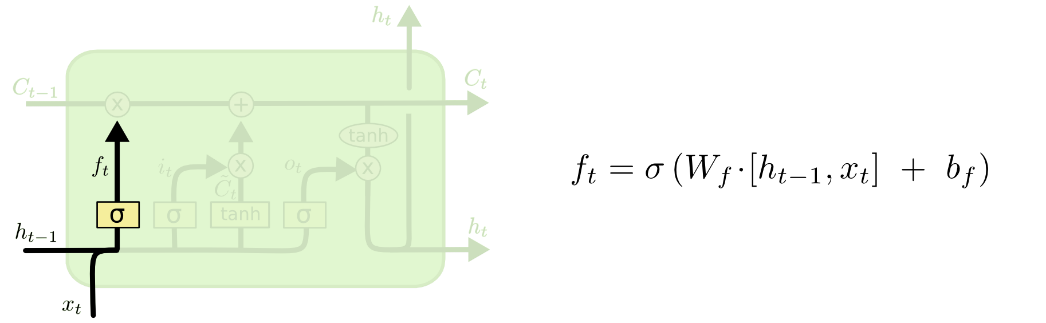


LSTM juga memiliki struktur seperti rantai ini, tetapi modul berulang memiliki struktur yang berbeda. Alih-alih memiliki satu lapisan jaringan saraf, ada empat, berinteraksi dengan cara yang sangat istimewa.



Kunci LSTM adalah cell state, garis horizontal yang melewati bagian atas diagram. Cell state ini berjalan lurus ke bawah seluruh rantai, dengan hanya beberapa interaksi linier kecil. Sangat mudah bagi informasi untuk mengalir begitu saja tanpa berubah. LSTM memiliki kemampuan untuk menghapus atau menambahkan informasi ke cell state, diatur oleh struktur yang disebut Gate. Gate adalah cara opsional untuk membiarkan informasi lewat. Mereka terdiri dari lapisan jaring saraf sigmoid dan operasi perkalian pointwise.

Alur kerja dari LSTM sendiri akan berjalan seperti berikut:



Langkah pertama adalah menentukan informasi apa yang akan dibuang dari cell state yang dilakukan oleh forget gate.

Diagram

Description automatically generated

output dari gerbang sigmoid, sehingga kami hanya menampilkan bagian yang kami putuskan.

Diagram

Description automatically generated with medium confidence

Melakukan update terhadap cell state yang sebelumnya dengan cell state yang baru /melakukan kolaborasi nilai yang baru

A picture containing text, clock

Description automatically generated

Gambar output dari cell state yang baru

Menentukan output yang yang berdasarkan oleh cell state yang telah difilter. Pertama, akan menjalankan lapisan sigmoid yang akan menentukan cell state mana yang akan menjadi output, setelah itu cell state akan melewati tanh yang akan mendorong nilai menjadi diantara -1 sampai 1 dan mengkalikan dengan output dari gerbang sigmoid, sehingga kami hanya menampilkan bagian yang diputuskan.