KLASIFIKASI NOMINAL UANG KERTAS RUPIAH TAHUN EMISI 2017 DENGAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* MENGGUNAKAN MXNET

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana
Program Studi Statistika

Disusun Oleh:

Ridha Nur Izah

14 611 179

JURUSAN STATISTIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2018

HALAMAN PERSETUJUAN PEMBIMBING

TUGAS AKHIR

Judul : Klasifikasi Nominal Uang Kertas Rupiah Tahun Emisi 2017 Dengan Algoritma *Convolutional Neural Network* Menggunakan MXNet

Nama : Ridha Nur Izah

Nomor Mahasiswa : 14 611 179

TUGAS AKHIR INI TELAH DIPERIKSA DAN DISETUJUI UNTUK DIUJIKAN

Yogyakarta, 16 Maret 2018

Mengetahui,

Dosen Pembimbing

Dr. RB Fajriya Hakim, M.Si

HALAMAN PENGESAHAN

TUGAS AKHIR

KLASIFIKASI NOMINAL UANG KERTAS RUPIAH TAHUN EMISI 2017 DENGAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* MENGGUNAKAN MXNET

Nama Mahasiswa : Ridha Nur Izah

Nomor Mahasiswa : 14 611 179

TUGAS AKHIR INI TELAH DIUJIKAN

18 APRIL 2018

Nama Penguji :	**Tanda Tangan**
1. Andrie Pasca Hendradewa, S.T., M.T	.....
2. Ayundyah Kesumawati, S.Si., M.Si	.....
3. Dr. RB Fajriya Hakim, S.Si., M.Si	.....

Mengetahui,

Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Drs. Allwar, M.Sc., Ph.D.

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Assalamu'laikum Warahmatullaahi Wabarakaatuh.

Puji syukur senantiasa penulis panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga tugas akhir yang berjudul **"Klasifikasi Nominal Uang Kertas Rupiah Tahun Emisi 2017 Dengan Algoritma *Convolutional Neural Network* Menggunakan MXNet"** dapat diselesaikan. Shalawat serta salam semoga selalu tercurah kepada Nabi Muhammad SAW serta para sahabat dan pengikutnya sampai akhir zaman.

Tugas akhir ini disusun sebagai salah satu persyaratan yang harus dipenuhi dalam menyelesaikan jenjang Strata Satu atau S1 di Jurusan Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia. Penyelesaian tugas akhir ini tidak terlepas dari bantuan, arahan, dan bimbingan dari berbagai pihak. Untuk itu penulis mengucapkan terima kasih kepada:

1. Allah *Subhanahu Wa Ta'ala* dan Rasulullah *Shallallahu'alaihi Wa Salam* yang telah memberikan nikmat iman islam serta karunia-Nya kepada penulis sehingga diberikan kelancaran dalam menyusun tugas akhir ini.
2. Kedua orang tua yang sangat saya cintai, Bapak Saidil Mustar dan Ibu Azmawati yang telah memberikan kasih sayang, kesabaran, doa, dukungan serta dukungan moril maupun materil.
3. Kedua kakak saya, Dio dan Adelina yang telah memberikan masukan-masukan dan pendapat dalam penyelesaian skripsi ini, do'a serta dukungan moril maupun materil.
4. Bapak Nandang Sutrisno, SH., LL.M., M.Hum., Ph.D., selaku rektor Universitas Islam Indonesia.
5. Bapak Drs. Allwar, M.Sc., Ph.D selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Islam Indonesia.

6. Bapak Dr. Raden Bagus Fajriya Hakim, S.Si., M.Si selaku Ketua Jurusan Statistika sekaligus dosen pembimbing skripsi, yang selalu mengingatkan serta membimbing penulis dan sangat berjasa dalam penyelesaian tugas akhir ini.
7. Dosen-dosen Statistika UII yang telah membina dan mendedikasikan ilmunya untuk penulis.
8. Keluarga besar Statistika FMIPA UII, para sesepuh, *master*, teman-teman dan adik-adik yang tidak bisa saya sebutkan satu persatu.
9. Teman-teman bimbingan "Sukses TA 2k18" yang dengan sabar mengajari penulis dan selalu menjawab pertanyaan-pertanyaan penulis.
10. Sahabat hidup di Jogja, "Mentari Club" dan Teman dekat penulis, yang sudah ikhlas merelakan waktunya, berbagi cerita dan pengalaman, memotivasi dan membuat kehidupan anak rantau menjadi indah selama di Jogja. Sukses untuk kita semua *gengs.*
11. Sanak-sanak Curup, yang selalu ada disisi penulis. Semoga kuliah kalian cepat rampung sanak dan selalu berada di dalam perlindungan-Nya.
12. Semua pihak lainnya yang tidak bisa penulis disebutkan satu per satu, terima kasih atas segala bantuannya.

Semoga segala bantuan, bimbingan, dan pengajaran yang telah diberikan kepada penulis mendapatkan imbalan dari Allah SWT. Penulis mohon maaf apabila selama proses penyusunan tugas akhir ini terdapat kekhilafan dan kesalahan. Penulis menyadari sepenuhnya akan keterbatasan kemampuan dalam penulisan tugas akhir ini, oleh karena itu penulis mengharap adanya kritik dan saran yang membangun demi kesempurnaan penyusunan dan penulisan tugas akhir ini. Semoga tugas akhir ini dapat bermanfaat bagi semua yang membaca dan membutuhkan.

Wassalamu'alaikum Warahmatullaahi Wabarakaatuh.

Yogyakarta, 16 Maret 2018

Ridha Nur Izah

DAFTAR ISI

HALAMAN JUDUL ..... **i**
HALAMAN PERSETUJUAN PEMBIMBING ..... **ii**
HALAMAN ..... **iii**
KATA PENGANTAR ..... **iv**
DAFTAR ISI ..... **vi**
DAFTAR TABEL ..... **viii**
DAFTAR GAMBAR ..... **ix**
DAFTAR LAMPIRAN ..... **x**
PERNYATAAN ..... **xi**
INTISARI ..... **xii**
ABSTRACT ..... **xiii**

BAB I PENDAHULUAN **1**
1.1 Latar Belakang Masalah 1
1.2 Rumusan Masalah 4
1.3 Batasan Masalah 4
1.4 Tujuan Penulisan 5
1.5 Manfaat Penulisan 5

BAB II TINJAUAN PERUSAHAAN **6**
BAB III LANDASAN TEORI **10**
3.1 Uang Kertas Rupiah 10
3.2 Klasifikasi 11
3.3 Citra 12
3.3.1 Definisi Citra 12
3.3.2 Representasi Citra Digital 12
3.3.3 Citra Warna 13
3.3.4 Citra *Grayscale* 14
3.3.5 Pengolahan Citra Digital 14
3.4 *Machine Learning* 16
3.5 *Deep Learning* 17
3.6 Pembelajaran *Backpropagation* 19
3.7 *Convolutional Neural Network* 20

3.7.1 Lapisan Konvolusi 22
3.7.2 *Fully-Connected Layer* 26
3.7.3 *Dropout Regularization* 27
3.7.4 *Softmax Classifier* 27
3.7 MXNet 28
BAB IV METODOLOGI PENELITIAN 29
4.1 Populasi dan Sampel Penelitian 29
4.2 Variabel dan Definisi Operasional 29
4.3 Sumber Data 30
4.4 Metode Analisis Data 30
4.5 Tahapan Penelitian 30
BAB V PEMBAHASAN 32
5.1 Pengumpulan Data Citra 32
5.2 *Preprocessing Citra* 34
5.3 Pembuatan Data *Set* 35
5.4 Proses Klasifikasi CNN 37
5.5 Parameter Pengujian 40
5.6 Hasil dan Pembahasan 42
5.6.1 Hasil *Preprocessing* 42
5.6.2 Hasil Pengujian Parameter 42
5.7 Hasil Klasifikasi CNN 46
BAB VI PENUTUP 48
6.1 Kesimpulan 48
6.2 Saran 48
DAFTAR PUSTAKA 50
LAMPIRAN 53

DAFTAR TABEL

Nomor **Judul** **Halaman**

Tabel 4.1 Definisi operasional variabel 29

Tabel 5.1 Perbandingan ukuran kernel 41

Tabel 5.2 Nilai parameter yang tidak berubah selama pengujian 41

Tabel 5.3 Hasil pengujian ukuran kernel 43

Tabel 5.4 Hasil pengujian jumlah *epoch*..... 44

Tabel 5.5 Hasil pengujian jumlah data..... 45

Tabel 5.6 Hasil klasifikasi data *train* 47

Tabel 5.5 Hasil klasifikasi data *test*..... 47

DAFTAR GAMBAR

Nomor **Judul** **Halaman**

Gambar 3.1 Citra uang kertas rupiah 2017 11
Gambar 3.2 Representasi citra digital dalam bentuk matriks 13
Gambar 3.3 Contoh citra digital 13
Gambar 3.4 Contoh citra warna 14
Gambar 3.5 Contoh Citra *Grayscale* 14
Gambar 3.6 Konversi citra warna ke *grayscale* 15
Gambar 3.7 Proses *resize* 16
Gambar 3.8 *Layer* pada *deep learning* 18
Gambar 3.9 Arsitektur jaringan *backpropagation* 19
Gambar 3.10 Struktur dari CNN untuk citra 21
Gambar 3.11 Proses konvolusi pada *input array* 23
Gambar 3.12 Interaksi proses konvolusi dan *neural network* 24
Gambar 3.13 Fungsi tanh 25
Gambar 3.14 Proses *max-pooling* 26
Gambar 3.15 Pengaplikasian *dropout regularization* 27
Gambar 4.1 *Flow chart* 31
Gambar 5.1 *Iphone* 6s 32
Gambar 5.2 Pecahan uang kertas sisi bagian depan 33
Gambar 5.3 Pecahan uang kertas sisi bagian belakang 33
Gambar 5.4 *Script* tahap *preprocessing* 34
Gambar 5.5 Memanggil data *set* 36
Gambar 5.6 *Script* untuk membangun datasets 36
Gambar 5.7 *Script* untuk pelatihan CNN 39
Gambar 5.8 Pembuatan dan prediksi model CNN 40
Gambar 5.9 *Output* proses *preprocessing* 42
Gambar 5.10 Grafik akurasi pengujian kernel 43
Gambar 5.11 Grafik akurasi pengujian *epoch* 45
Gambar 5.12 Grafik akurasi pengujian jumlah data 46

DAFTAR LAMPIRAN

Lampiran 1 : *Script Convolutional Neural Network* Menggunakan MXNet

PERNYATAAN

Dengan ini penulis menyatakan bahwa dalam tugas akhir ini tidak terdapat karya yang sebelumnya pernah diajukan untuk memperoleh gelar kesarjanaan di suatu perguruan tinggi dan sepengetahuan penulis juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 18 April 2018



Ridha Nur Izah

KLASIFIKASI NOMINAL UANG KERTAS RUPIAH TAHUN EMISI 2017 DENGAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* MENGGUNAKAN MXNET

Ridha Nur Izah

Program Studi Statistika Fakultas MIPA

Universitas Islam Indonesia

INTISARI

Penerapan teknologi otomasi akan mempermudah aktivitas manusia. Contoh penerapan teknologi otomasi adalah pada proses jual beli dengan mesin sebagai perantaranya. Mesin tersebut bertindak sebagai penjual yang cara kerjanya seperti otak manusia, yaitu harus bisa membaca dan mengenali setiap nominal uang secara cepat dan tepat. Metode CNN adalah salah satu metode *deep learning* yang mampu melakukan proses ekstraksi fitur dan klasifikasi. Metode ini merupakan *neural network* yang didahului dengan lapisan konvolusi. Lapisan konvolusi sendiri terdiri dari proses konvolusi dan *pooling*. Pada penelitian ini, diterapkan algoritma *convolutional neural network* (CNN) untuk mengklasfikasikan nominal uang kertas. Proses yang dilakukan pada penelitian ini adalah pengumpulan data, proses *preprocessing*, klasifikasi dan pengujian. *Preprocessing* dilakukan dengan mengubah ukuran citra menjadi 32x32 dan mengubah citra menjadi *grayscale*. Klasifikasi dilakukan dengan CNN, dimana untuk mendapatkan akurasi terbaik parameter-parameter yang ada harus diujikan. Selain itu, percobaan juga dilakukan dengan menggunakan ukuran data latih yang beragam untuk melihat pengaruh jumlah data latih terhadap akurasi. Hasil penelitian menunjukkan bahwa ukuran kernel 3x3 untuk konvolusi pertama dan 2x2 untuk konvolusi kedua, jumlah filter 20, jumlah iterasi 50, dan jumlah neuron pada lapisan tersembunyi 17 akan menghasilkan akurasi terbaik sebesar 93,57%.

Kata Kunci: *Convolutional Neural Network*, *Deep Learning*, Klasifikasi Nominal Uang Kertas, Pengolahan Citra Digital

NOMINAL CLASSIFICATION OF BANKNOTES IN 2017 EY WITH CONVOLUTIONAL NEURAL NETWORK ALGORITHM USING MXNET

Ridha Nur Izah

Statistics Faculty of Mathematics and Natural Sciences

Islamic University of Indonesia

ABSTRACT

Automation technology will make human activities easier. The example of automation technology is sale process with machine as a mediator. The machine as a seller that works like a human brain, must be able to read and recognize every nominal of banknotes quickly and accurately. CNN is one of deep learning method that is able to perform automatic feature extraction and classification. This method is a neural network which is preceded by convolution layer. Convolution layer itself consists of convolution and pooling process. In this study, convolutional neural network (CNN) will be applied to classifying nominal of banknotes. This process involved in this study is data collection, preprocessing, classification and testing. Preprocessing is done by resizing image to 32x32 pixels and changing image to grayscale. Classification is done by CNN to get the best accuracy which existing parameters should be tested and evaluated. Moreover, the system will do the experiments which is using variety of training data size to see the effect of the amount of training data on accuracy.. The experiments result showed that kernel size 3x3 in the first convolution and 2x2 in the second convolution, 20 filters, 50 epoch and the number of neurons in the hidden layer 17 will achieve best accuracy of 93,57%.

Kata Kunci: *Convolutional Neural Network, Deep Learning, Image Processing, Nominal Classification of Banknotes*

BAB I
LATAR BELAKANG

1.1 Latar Belakang

Transaksi perdagangan yang memungkinkan adanya proses jual beli sudah berlangsung selama berabad-abad. Pada zaman dahulu, manusia menggunakan sistem barter sebagai alat pembayaran sebelum mereka mengenal uang. Seiring dengan berjalannya waktu transaksi jual beli sudah tidak menggunakan sistem barter tetapi menggunakan uang sebagai alat pembayaran yang sah.

Mata uang negara Indonesia adalah rupiah. Menurut Pasal 11 Undang-Undang Nomor 7 Tahun 2011 Bank Indonesia menjadi satu-satunya lembaga yang berwenang melakukan pengeluaran, pengedaran dan atau pencabutan rupiah. Setiap uang memiliki nilai nominal yang berguna untuk menentukan nilai dari suatu barang dan jasa yang diperjualbelikan. Ada 2 jenis uang, yaitu uang kartal dan uang giral. Uang kartal adalah uang yang digunakan secara langsung dalam proses tukar menukar barang atau jasa. Uang kartal terdiri dari dua jenis yaitu uang kertas dan uang logam.

Pada proses jual beli konvensional pastinya akan terjadi pertemuan antara penjual dan pembeli yang biasanya dilakukan di pasar, warung atau tempat-tempat umum lainnya. Namun dengan perkembangan teknologi sekarang ini maka proses jual beli dapat dilakukan tanpa adanya pertemuan langsung antara penjual dan pembeli. Sistem *online* adalah salah satu contoh proses jual beli yang tidak mempertemukan penjual dan pembeli secara langsung. Sistem *online* tersebut memanfaatkan teknologi internet sehingga transaksi jual beli dapat dilakukan melalui *handphone* atau komputer yang terhubung ke internet. Selain itu, ada proses jual beli yang menggunakan mesin sebagai perantaranya. Proses jual beli tersebut hanya melibatkan manusia sebagai pembeli dan mesin sebagai penjual dengan tetap menggunakan uang sebagai alat pembayarannya. Jika ingin mendukung proses jual beli antara manusia dengan mesin maka mesin sebagai

penjual harus bisa membaca dan mengenali setiap nominal uang seperti yang dilakukan oleh manusia.

Berdasarkan hal tersebut maka dibutuhkan suatu sistem yang cara kerjanya seperti otak manusia. Sistem itu harus dapat mengenali dan mengklasifikasikan nominal uang kertas secara cepat dan tepat seperti yang dilakukan oleh manusia. Sistem tersebut akan digunakan pada mesin otomatis. Jika mesin menerima salah satu pecahan uang kertas, mesin itu dapat langsung mengenali nominal dari uang tersebut. Sehingga sistem ini dapat dikembangkan dan digunakan sebagai alat bantu penjualan yang melibatkan mesin. Proses pengenalan uang kertas tersebut dilihat dari gambar permukaannya atau pengenalan citra *(image recogniton).* Klasifikasi adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. "Proses klasifikasi biasanya dibagi menjadi dua fase yaitu fase *training* dan fase *testing*. Pada fase *training*, sebagian data yang telah diketahui kelas datanya diumpankan untuk membentuk model perkiraan. Kemudian pada fase *testing* model yang sudah terbentuk diuji dengan sebagian data lainnya untuk mengetahui akurasi dari model tersebut, bila akurasinya mencukupi model ini dapat dipakai untuk prediksi kelas data yang belum diketahui" (Pramudiono, 2003).

Bidang teknologi yang dapat merealisasikan hal tersebut adalah *Artificial Intelligence* atau AI. Menurut Winston dan Prendergast (1984), tujuan membuat AI ada 3, yaitu:

1. Membuat mesin menjadi lebih pintar (tujuan utama)
2. Memahami apa itu kecerdasan (tujuan ilmiah).
3. Membuat mesin lebih bermanfaat (tujuan entrepreneurial)

Pengaplikasian AI ini tidak terlepas dari *Machine Learning,* karena secanggih apapun mesin, untuk melakukan tugas secara otomatis memerlukan pembelajaran dan uji coba. Pada penelitian ini, teknik yang digunakan untuk mengimplementasikan AI adalah *Deep Learning. Deep Learning* merupakan

perkembangan dari *Machine Learning*. *"Deep Learning* adalah tentang belajar beberapa tingkat representasi dan abtraksi yang membantu untuk memahami data seperti gambar, suara dan teks" (LISA lab, 2015). *Deep Learning* akan membuat jaringan neuron buatan yang dirancang secara berlapis-lapis untuk melakukan klasifikasi perintah yang di*input* hingga akhirnya menghasilkan *output.* Salah satu algoritma *Deep Learning* yang menjadi *trend* saat ini dan dapat diterapkan untuk mengklasifikasikan data citra adalah *Convolutional Neural Network* (CNN). Algoritma *Convolutional Neural Network* (CNN) dilatih untuk mencari fitur seperti tepi, sudut, perbedaan warna dan menggabungkannya menjadi bentuk yang lebih kompleks.

Contoh penelitian dengan studi kasus klasifikasi nominal uang kertas telah dilakukan oleh Wicaksono (2007) dengan menggunakan metode Jaringan Syaraf Tiruan *Backpropagation*. Hasil dari penelitian tersebut adalah sistem dapat mengklasifikasi nominal uang kertas menjadi beberapa kelas. Ciri dari setiap uang kertas memiliki pola yang unik dan ciri-ciri tersebut berbeda antara satu dengan lainnya, baik bentuk angka, jumlah angka nol serta gambar sisi depan atau belakangnya. Pola khusus inilah yang akan dilatih dalam sistem Jaringan Syaraf Tiruan layaknya pembelajaran otak manusia untuk membedakan nominal uang kertas.

Selain itu, contoh penelitian yang menggunakan metode *Convolutional Neural Network* (CNN) dengan judul "Implementasi *Convolutional Neural Network* Untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital" oleh Rismiyati (2016). Pengujian model yang digunakan pada penelitian ini adalah dengan metode *stratified cross validation.* Pengujian dilakukan untuk dua macam model, yaitu model 2 kelas dan model 4 kelas. Pada model dua kelas digunakan 1 lapisan konvolusi, 15 *fiter* (ukuran 3x3x3) dan 100 neuron di *hidden layer* didapatkan akurasi sebesar 81,45%. Sedangkan untuk model 4 kelas digunakan 2 lapisan konvolusi, 6 filter (ukuran 3x3) dan 150 neuron pada *hidden layer* menghasilkan

akurasi sebesar 70,7%. Kemudian jumlah data latih yang digunakan juga ditambah menjadi 875, sehingga akurasi yang didapatkan sebesar 81,45%.

Berdasarkan dari dua penelitian yang telah dijelaskan sebelumnya, harapan penulis metode *Deep Learning* dengan algoritma *Convolutional Neural Network* (CNN) juga dapat mengklasifikasikan nominal uang kertas rupiah dengan *input* citra uang kertas rupiah tahun emisi 2017. Oleh karena itu, penulis membuat penelitian yang berjudul **"Klasifikasi Nominal Uang Kertas Rupiah Tahun Emisi 2017 Dengan Algortima *Convolutional Neural Network* Menggunakan MXNet".** Alasan menggunakan *framework* MXNet adalah karena MXNet merupakan *framework* yang sederhana dan juga MXNet dapat membangun arsitektur jaringan dari awal yang dapat di kustom sedemikian rupa untuk menghasilkan akurasi yang baik. Serta pelatihan data lebih cepat dan tidak membutuhkan banyak *space memory.*

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka dapat dirumuskan masalah sebagai berikut:

1. Bagaimana mengklasifikasi nominal uang kertas dengan menggunakan algoritma *convolutional neural network* Menggunakan MXNet?
2. Bagaimana tingkat akurasi model algoritma *convolutional neural network* terhadap klasifikasi nominal uang kertas?

1.3 Batasan Masalah

Permasalahan yang dibahas pada penelitian ini memiliki ruang lingkup yang luas, sehingga diberikan batasan-batasan sebagai berikut:

1. Citra yang digunakan adalah citra uang kertas rupiah tahun emisi 2017 dengan pecahan Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00.
2. Jumlah data citra yang digunakan adalah 700 citra dengan jumlah 100 citra pada masing-masing pecahannya.

3. Data citra yang digunakan adalah data primer yang diambil menggunakan kamera utama *Iphone* 6s.
4. Algoritma yang digunakan adalah *Convolutional Neural Netwok.*
5. Alat bantu yang digunakan adalah RStudio versi 1.1.383 dengan *framework* MXNet.

1.4 Tujuan Penelitian

Tujuan yang akan dicapai dalam penelitian ini antara lain:

1. Membangun sistem yang dapat mengklasifikasikan nominal uang kertas dengan algoritma *convolutional neural network* menggunakan MXNet.
2. Memperoleh tingkat akurasi hasil klasifikasi nominal uang kertas.

1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut:

1. Menerapkan dan mengembangkan pengetahuan tentang *convolutional neural network* untuk mengklasifikasikan nominal uang kertas.
2. Memberikan kontribusi bagi perkembangan pengolahan citra digital dalam bidang perekonomian khususnya dalam klasifikasi nominal uang kertas.
3. Membangun sistem yang dapat dikembangkan dan digunakan sebagai alat bantu penjualan yang melibatkan mesin.

BAB II
TINJAUAN PUSTAKA

Penelitian yang berkaitan dengan algoritma *Convolutional Neural Network* (CNN) pada umumnya telah banyak dilakukan oleh beberapa peneliti-peneliti sebelumnya. Serta penelitian mengenai klasifikasi nominal uang kertas juga sudah banyak diteliti dengan menggunakan metode Jaringan Syaraf Tiruan. Berikut beberapa penelitian yang berhubungan dengan penelitian ini yang telah dilakukan oleh peneliti sebelumnya.

Salah satu penelitian yang berhubungan dengan algoritma *Convolutional Neural Network* adalah penelitian yang dilakukan oleh Zufar (2016) dengan judul "*Convolutional Neural Networks* Untuk Pengenalan Wajah Secara *Real-Time*". Penelitian tersebut menggunakan 7 *layer* model konvolusi dan data *train* sebanyak 126 citra. Kesimpulan dari penelitian yang dilakukan Zulfar adalah kinerja pengenalan wajah mendapat rata-rata tingkat akurasi lebih dari 89% dalam ± 2 *frame* per detik. Tingkat akurasi tersebut diperoleh dari kontruksi model CNN sampai ke-7 lapisan dari input hasil ekstraksi *extended local binary pattern* dengan radius 1 dan *neighbor* 15.

Danukusumo (2017) dalam tugas akhirnya yang berjudul "Implementasi *Deep Learning* Menggunakan *Convolutional Neural Network* Untuk Klasifikasi Citra Candi Berbasis GPU". Hasil dari penelitian ini adalah akurasi sebesar 98,99% dan akurasi untuk data *testing* sebesar 85,57%. Kedua akurasi tersebut terbilang tinggi karena lebih dari 80% model dapat mengklasifikasikan citra candi. Dari kedua akurasi tersebut dapat ditarik kesimpulan bahwa teknik *Deep Learning* menggunakan metode CNN dapat mengklasifikasikan citra candi dengan sangat baik.

Putra (2016) dalam tesis yang berjudul "Klasifikasi Citra Menggunakan *Convolutional Neural Network (CNN)* pada *Caltech* 101" adalah penelitian yang menggunakan metode CNN juga. Ada 5 kategori unggas yang digunakan pada penelitian ini, yaitu *emu, flamingo, ibis, pigeon* dan *rooster* dengan jumlah citra

sebanyak 150 citra. Selain kategori tersebut terdapat 3 kategori lainnya, yaitu cougar, crocodile dan face. Kesimpulan dari klasifikasi 5 kategori unggas mendapatkan presentase keberhasilan sebesar 20% dengan nilai *error training* sebesar 0,1224 dan kesimpulan dari klasifikasi 5 kategori unggas mendapatkan presentase keberhasilan sebesar 50% dengan nilai *error training* sebesar 0,0171.

Penelitian yang masih berkaitan dengan algoritma *Convolutional Neural Network* adalah penelitian yang dilakukan oleh Krizhevsky dkk (2015) dengan judul "*ImageNet Classification with Deep Convolutional Neural Networks".* Penelitian tersebut menyatakan bahwa jaringan syaraf tiruan tersebut memiliki 60 juta parameter dan 650 ribu neuron serta terdiri dari lima lapisan konvolusi. Untuk membuat *training* lebih cepat maka mereka menerapkan GPU yang sangat efisien dari operasi konvolusi untuk mengurangi *overfitting.* Dari hasil uji coba didapatkan tingkat kesalahan top-5 sebesar 15.3% dan top-2 sebesar 26.2%.

Selain penelitian yang berkaitan dengan metode yang digunakan terdapat penelitian lainnya yang juga berkaitan dengan klasifikasi nominal uang kertas yang dilakukan oleh Handayani (2014) yang berjudul "Penggunaan Algoritma *Backpropagation Levenberg Marquardt* dan Teknik Pengolahan Citra Digital Untuk Identifikasi Nominal Uang Kertas". Tingkat akurasi yang didapatkan dengan menggunakan Algoritma *Backpropagation Levenberg Marquardt* untuk mendeteksi nominal uang kertas sebesar 50%. Tingkat akurasi tersebut termasuk rendah jika dibandingkan dengan penelitian sebelumnya dan menggunakan metode jaringan syaraf tiruan lainnya.

Anindita dkk (2012) dalam jurnal ICT yang berjudul "Aplikasi Pendeteksi dan Pengenalan Nominal Uang Kertas Dengan Metode *Learning Vector Quantization* Sebagai Alat bantu Bagi Penyandang Tuna Netra". Dengan menggunakan 50 *epoch* akurasi data *training* sebesar 100% dan akurasi data *testing* sebesar 83,39%. Dengan menggunakan 100 *epoch* akurasi data *training* sebesar 100% dan akurasi data *testing* sebesar 83,39%. Dengan menggunakan 200 *epoch* akurasi data *training* sebesar 85,71% dan akurasi data *testing* sebesar 78,78%. Tingkat akurasi yang dihasilkan dari pengujian seluruh nominal uang adalah sebesar 89,39%. Penelitian tersebut juga membandingkan nilai *learning*

rate. *Learning rate* 0,09 menghasilkan akurasi sebesar 100% dan *learning rate* 0,05 menghasilkan akurasi sebesar 85,71%. Selain *epoch* dan *learning rate* hal lain yang dibandingkan adalah *node hidden layer*. Dengan menggunakan 32 *node hidden layer* akurasi yang didapatkan sebesar 100% sedangkan jika menggunakan 64, 100 dan 176 *node hidden layer* akurasi yang didapatkan sebesar 85,71%. Jadi, kesimpulannya dengan menggunakan 50 *epoch*, *learning rate* 0,09 dan *node hidden layer* 32 akan menghasilkan akurasi terbesar, yaitu 89,39% dengan rata-rata waktu komputasi pengujian adalah 0,151 detik.

Penelitian yang berjudul "Identifikasi Uang Kertas Menggunakan Metode *Backpropagation* Pada Sistem Internet-Telepon Umum" oleh Kharisma dan Achmad Subhan (2011) merupakan penelitian mengenai identifikasi uang kertas dengan menggunakan metode jaringan syaraf tiruan lainnya. Proses pengujian yang digunakan adalah *learning rate* dengan nilai 0,1. Hasil dari pengujian mendapatkan tingkat kebenaran proses identifikasi sistem hingga sebesar 96,85% dan tingkat sensitifitas sistem dari identifikasi uang kertas berbeda-beda. Pada uang kertas Rp 1.000,00 sensitifitas sistem mencapai 86,66%, pada uang kertas Rp 2.000,00 sensitifitas sistem mencapai 76,92%, sedangkan pada uang kertas Rp 5.000,00 sensitifitas sistem mencapai 62,50%.

Penelitian lainnya mengenai deteksi nominal uang kertas adalah penelitian yang dilakukan oleh Saputra dkk (2016) dengan judul "Implementasi Metode Jaringan Syaraf Tiruan (JST) Pada Alat Deteksi Nilai Nominal Uang". Penelitian tersebut merancang sebuah alat bantu untuk mendeteksi nominal uang kertas. Alat pendeteksi itu akan digunakan oleh penyandang tuna netra untuk membantu aktifitas jual beli yang dilakukan setiap hari dan berulang-ulang secara efektif dan efisien. Dari hasil 10 kali pengujian alat menggunakan uang pecahan Rp 10.000,00 didapatkan hasil 7 kali benar dan 3 kali salah sehingga dapat dikatakan akurasi pengenalan citra sebesar 70%. Hasil 10 kali pengujian alat menggunakan uang pecahan Rp 20.000,00 didapatkan hasil hanya 1 kali salah sehingga dapat dikatakan akurasi pengenalan citra sebesar 90%. Hasil 10 kali pengujian alat menggunakan uang pecahan Rp 50.000,00 didapatkan hasil hanya 1 kali salah sehingga dapat dikatakan akurasi pengenalan citra sebesar 90%. Hasil

10 kali pengujian alat menggunakan uang pecahan Rp 100.000,00 didapatkan hasil semua benar sehingga dapat dikatakan akurasi pengenalan citra sebesar 100%. Kemudian peneliti mengkombinasikan keempat pecahan uang tersebut dengan 20 kali pengujian, akurasi yang dihasilkan sebesar 90%.

BAB III
LANDASAN TEORI

3.1 Uang Kertas Rupiah

Uang adalah segala sesuatu yang siap sedia dan pada umumnya diterima dalam pembayaran pembelian barang-barang, jasa-jasa dan untuk membayar utang. Rupiah adalah mata uang yang dikeluarkan oleh Negara Kesatuan Republik Indonesia, UU No. 23 tahun 1999 melindungi penggunaan rupiah. Bank Indonesia menjadi satu-satunya lembaga yang berwenang melakukan pengeluaran, pengedaran dan atau pencabutan rupiah. Uang merupakan alat yang paling penting dalam melakukan transaksi yang digunakan oleh manusia. Setiap uang memiliki nilai nominal yang berguna untuk menentukan nilai dari suatu barang dan jasa yang diperjualbelikan.

Uang kertas yang ada pada masa sekarang adalah kertas dengan gambar tertentu dengan nilai nominal ditentukan secara arbitrer oleh negara lewat keputusan politik. Uang kertas muncul pertama kali di Cina. Penemu uang kertas yang berasal dari Cina adalah Ts'ai Lun. Lun konon membuat uang kertas pertama dari kulit kayu pohon murbei yang daunnya sebagai pakan ulat untuk industri sutra Cina. Setelah itu, Cina mulai memproduksi uang kertas pertama di dunia. Bangsa lain mulai menggunakan uang kertas pertama setelah Marcopolo singgah di Cina pada abad ke-13. Umumnya masyarakat percaya pada mata uang ini, sekalipun bendanya dibuat dari kertas yang sama sekali nilainya sangat jauh di bawah nilai emas. Karena atas dasar kepercayaan itulah maka disebut juga sebagai uang kepercayaan

Uang kertas yang beredar pada zaman ini dikenal sebagai *fiat money* dan *token money*. Maksud dari *fiat money* adalah alat tukar dari kertas dan tidak didukung oleh komoditas apapun. Sedangkan maksud dari *token money*, yaitu uang yang tidak bernilai penuh atau uang yang nilai intrinsiknya lebih kecil dari pada nominalnya.

Uang kertas rupiah tahun emisi 2017 yang akan digunakan dalam

penelitian ini, dengan pecahan Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00 pada sisi depan (gambar pahlawan nasional Indonesia) serta sisi belakang (tarian tradisional Indonesia). Berikut adalah gambar uang kertas rupiah tersebut:



Gambar 3.1 Citra uang kertas rupiah 2017

3.2 Klasifikasi

Klasifikasi adalah bentuk analisis data yang mengekstrak model yang menggambarkan kelas data penting. Misalnya, kita akan membangun model klasifikasi untuk mengkategorikan pinjaman bank ke dalam dua kelas yaitu sebagai kelas aman atau kelas berisiko, analisis semacam itu dapat membantu kita untuk lebih memahami data secara lebih luas. Klasifikasi terdiri dari dua tahapan proses. Tahap pertama disebut juga sebagai tahap pembelajaran (di mana model klasifikasi dikembangkan) dan tahap kedua adalah tahap klasifikasi, di mana model digunakan untuk memprediksi label kelas untuk data yang diberikan.

"Keakuratan suatu model klasifikasi dapat diketahui dari seperangkat tes yang diberikan dengan menghitung persentase dari data uji yang diklasifikasikan dengan benar oleh model klasifikasi, jika keakuratan klasifikasi dianggap dapat diterima, klasifikasi dapat digunakan untuk mengklasifikasikan data lainnya yang label kelasnya tidak diketahui" (Han dan Kamber, 2006).

Setiap teknik dari klasifikasi tersebut terdapat kekurangan dan kelebihan. "Tetapi prinsip dari masing-masing teknik tersebut sama, yaitu melakukan suatu

pelatihan sehingga di akhir pelatihan, model dapat memprediksi setiap vektor masukan ke label kelas *output* dengan tepat" (Prasetyo, 2012). Mengklasifikasi citra uang kertas rupiah adalah salah satu cara untuk mengorganisasikan citra-citra tersebut. Citra yang memiliki kesamaan gambar, angka, tulisan dan lainnya akan dikelompokkan ke dalam kategori yang sama.

3.3 Citra

3.3.1 Definisi Citra

"Citra bisa didefinisikan sebagai suatu fungsi dua dimensi, f(x,y), dimana x dan y adalah kooordinat spasial, dan f(x,y) adalah nilai pada koordinat (x,y) yang sering disebut intensitas. Citra digital adalah citra f(x,y) yang telah didigitalisasi baik dari segi koordinat area maupun pada nilai intensitasnya. Sebuah citra digital terdiri dari sejumlah elemen, yang masing-masing mempunyai lokasi dan nilai tertentu. Masing-masing elemen disebut *picture elements atau pixel*" (Gonzales, Rafael C., Richard E. Woods dan Steven L. Eddins, 2008). Setiap titik pada citra dapat dinyatakan dengan:

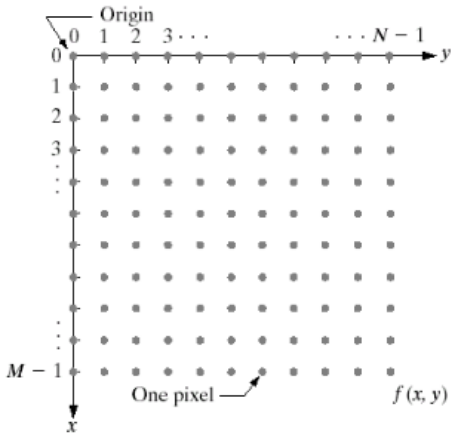
$$0 < f(x,y) < \infty \quad (1)$$

dimana:

- f adalah intensitas cahaya pada koordinat (x,y)
- (x,y) adalah perpotongan garis antara x dan y

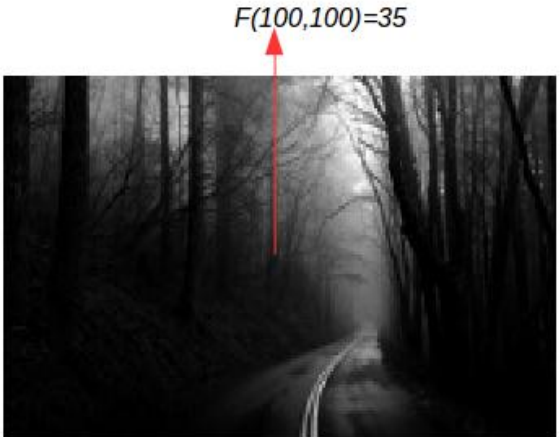
3.3.2 Representasi Citra Digital

Nilai *pixel* didapatkan dari dua buah bilangan bulat dan bilangan bula tersebut menunjukkan lokasi bidang citra di dalam komputer. Contohnya, pojok kiri atas citra direpresentasikan pada koordinat (0,0) atau pojok kanan bawah citra direpresentasikan pada koordinat (m-1, n-1), citra-citra tersebut berukuran m x n *pixel*. Cara lain untuk merepresentasikan citra digital adalah dengan menggunakan matriks. Citra digital tersebut merupakan matriks dua dimensi dengan jumlah N kolom dan jumlah M baris. Nilai intensitas *pixel* adalah sebuah elemen matriks dua dimensi tersebut. Berikut adalah gambaran dari matriks citra digital:



Gambar 3.2 Representasi citra digital dalam bentuk matriks

Gambar 3.3 menunjukkan contoh citra digital yang berukuran 200x200 *pixel*. Pada gambar tersebut ditunjukkan nilai intensitas pixel pada koordinat x=100 dan y=100 adalah 35.



Gambar 3.3 Contoh citra digital

3.3.3 Citra Warna

Citra warna merupakan citra yang titiknya mempunyai warna yang spesifik dan merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau dan biru. Sehingga format citra warna ini sering disebut sebagai citra RGB *(red-green-blue).* Citra warna terdiri dari tiga *keeping* warna yang masing-masing menyatakan intensitas warna merah, warna hijau dan warna biru, untuk setiap *keeping* intensitas dengan nilai maksimum 255 (8 bit). Berikut merupakan contoh citra warna:



Gambar 3.4 Contoh citra warna

3.3.4 Citra *Grayscale*

Citra *grayscale* dikenal juga sebagai citra berskala keabuan. Citra jenis ini menangani gradasi hitam dan putih yang menghasilkan efek keabuan. Skala pada citra *grayscale* bergantung pada jumlah bit yang digunakan. Untuk citra berskala 8 bit dinyatakan dengan intensitas dengan skala 0 sampai 255. Nilai 0 menyatakan hitam dan 255 menyatakan putih. Untuk nilai di antara 0 sampai 255 memiliki warna keabuan. Berikut merupakan contoh citra *grayscale*:



Gambar 3.5 Contoh citra *grayscale*

3.3.5 Pengolahan Citra Digital

Secara umum pengolahan citra memiliki definisi pengolahan gambar berdimensi dua melalui komputer digital. Citra diolah pada tiap *pixel* (x,y) untuk menghasilkan citra baru yang sesuai dengan kebutuhan. Citra adalah bagian dari komponen multimedia yang memegang pernanan penting dalam menyajikan informasi visual. Tahapan pertama sebelum mengolah citra adalah mengumpulkan citra itu sendiri. Salah satu cara pengumpulan citra yaitu, peneliti dapat mengambil foto citra secara langsung. Hal tersebut lebih efisien karena *background* dan sisi-sisi citra yang didapatkan akan sama. Selain itu, peneliti dapat mengambil citra disalah satu situs terbesar di dunia, yaitu *google.com.* Tahapan kedua untuk mengolah citra diginal adalah *preprocessing* citra. *Preprocessing* citra perlu dilakukan agar citra tersebut dapat diolah ketahapan selanjiynya, baik untuk kebutuhan ekstraksi fitur maupun kebutuhan klasifikasi. Ada beberapa tahap pada *preprocessing* citra, yaitu:

Grayscaling

Grayscaling citra adalah proses merubah skala citra dari citra berskala warna ke citra berskala keabuan *(grayscale).* Pada citra warna yang terdapat tiga kanal warna merah, hijau dan biru di konversi citra warna ke citra *grayscale* seperti pada persamaan (2), yaitu:

$$M(i,j) = (0{,}299 \text{ x } R) + (0{,}587 \text{ x } G) + (0{,}144 \text{ x } B) \qquad (2)$$

Pada rumus di atas, M merupakan hasil intensitas pada citra *grayscale* sedangkan nilai R, G dan B adalah nilai dari intensitas warna pada kanal merah, biru dan hijau *(red-green-blue)* citra warna. Sedangkan bobot yang dikalikan pada setiap kanal R, G dan B merupakan nilai yang ditetapkan secara proporsional respon presepsi mata manusia untuk masing-masing warna merah, hijau dan biru. Bobot standar telah ditentukan oleh NTSC *(National Television System Committee)* dengan nilai bobot merah sebesar 0,299, nilai bobot hijau sebesar 0,587 dan nilai bobot biru sebesar 0,114.



Gambar 3.6 Konversi citra warna ke *grayscale*

Pada **gambar 3.6** di atas, terdapat 2 buah citra uang Rp 20.000,00. Pada gambar di sebelah kiri merupakan citra warna dan di sebelah kanan adalah citra *grayscale* yang merupakan hasil dari proses *grayscaling* pada citra warna.

Resize

Resize citra adalah proses merubah ukuran jumlah *pixel* suatu citra digital. Pada penelitian ini proses *resize* dilakukan untuk menyamakan ukuran *pixel* pada semua data citra uang kertas. Menyamakan ukuran *pixel* berarti membuat data citra pada uang kertas dengan ukuran jumlah *pixel* yang sama antara baris dan kolomnya. Setelah semua citra digital disamakan pada jumlah baris dan kolomnya maka data citra siap untuk diproses pada tahap selanjutnya.



Gambar 3.7 Proses *resize*

Pada gambar di atas, terdapat dua buah citra *grayscale* uang Rp 20.000,00. Pada gambar di sebelah kiri merupakan citra dengan dimensi baris 226 *pixel* dan kolom 97 *pixel* dan di sebelah kanan merupakan citra hasil dari proses *resize* pada citra *grayscale* sebelah kiri dengan ukuran dimensi baris 150 *pixel* dan kolom 64 *pixel.*

Tahapan ketiga setelah *preprocessing* adalah membuat data *set.* Ada 2 data *set* yang biasanya digunakan, yaitu *training* data dan *testing* data. Pada proses *training,* data akan dilatih sehingga didapatkan model *training* sedangkan pada proses *testing,* model tersebut akan divalidasi. Tahapan terakhir adalah melakukan klasifikasi CNN.

3.4 *Machine Learning*

Arthur Samuel adalah orang yang pertama kali memperkenalkan istilah *Machine Learning. Machine learning* adalah salah satu bidang ilmu komputer yang memberikan kemampuan pembelajaran kepada komputer untuk mengetahui sesuatu tanpa pemrogram yang jelas. *Machine learning* dapat didefinisikan sebagai metode komputasi berdasarkan pengalaman untuk meningkatkan performa atau membuat prediksi yang akurat. Definisi pengalaman disini ialah informasi sebelumnya yang telah tersedia dan bisa dijadikan data pembelajar. Dalam pembelajaran *machine learning*, terdapat beberapa skenario-skenario sebagai berikut:

1. *Supervised Learning*

   Penggunaan skenario *supervised learning*, pembelajaran menggunakan masukan data pembelajaran yang telah diberi label. Setelah itu membuat prediksi dari data yang telah diberi label.

2. *Unsupervised Learning*

   Penggunaan skenario *unsupervised learning*, pembelajaran menggunakan masukan data pembelajaran yang tidak diberi label. Setelah itu mencoba untuk mengelompokan data berdasarkan karakteristik-karakteristik yang ditemui.

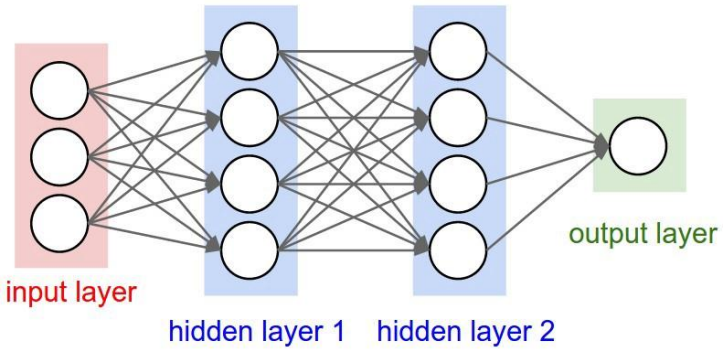
3. *Reinforcement Learning*

   Pada skenario *reinforcement learning* fase pembelajaran dan tes saling dicampur. Untuk mengumpulkan informasi pembelajar secara aktif dengan berinteraksi ke lingkungan sehingga untuk mendapatkan balasan untuk setiap aksi dari pembelajar.

3.5 *Deep Learning*

Deep Learning merupakan cabang ilmu dari *Machine Learning* berdasarkan satu *set* algoritma yang digunakan untuk model abstraksi tingkat tinggi pada data dengan menggunakan beberapa lapisan implementasi dan menggunakan struktur yang kompleks atau sebaliknya, terdiri dari beberapa transformasi *non-linear*. Dalam *Deep Learning*, sebuah komputer dapat belajar mengklasifikasikan gambar, suara, teks atau video secara langsung. Sebuah komputer seperti dilatih dengan menggunakan data *set* berlabel dan jumlahnya sangat besar yang kemudian dapat mengubah nilai piksel dari sebuah gambar menjadi suatu representasi internal atau *featur vector* yang dimana pengklasifikasiannya dapat digunakan untuk mendeteksi atau mengklasifikasi pola pada masukan *input* (LeCun, Bengio & Hinton, 2015).

Metode *deep learning* adalah metode pembelajaran dengan beberapa tingkat representasi, dimana representasi dapat membentuk arsitektur jaringan syaraf yang mempunyai banyak *layer*. *Layer* pada *deep learning* terbagi menjadi tiga bagian yaitu, *input layer*, *hidden layer* dan *output layer*. Pada *hidden layer* dapat dibuat banyak lapisan untuk menemukan komposisi algoritma yang tepat agar dapat meminimalisir *error* pada *output*. Semakin banyak *layer* yang dihasilkan, maka akan semakin kecil *error* yang dihasilkan. Sehingga dapat menghasilkan akurasi yang lebih baik.



Gambar 3.8 *Layer* pada *deep learning*

Pada **gambar 3.8** di atas merupakan contoh *layer* pada *deep learning*. Lingkaran berwarna putih menggambarkan neuron. Setiap lapisan *hidden layer* mempunyai satu atau lebih neuron. Neuron-neuron ini yang akan terhubung langsung dengan neuron lain pada *layer* berikutnya. Hubungan antar neuron hanya akan terjadi diantara 2 buah *layer (input* dan *output),* tidak ada hubungan pada *layer* yang sama walaupun jika secara teknis bisa saja dibuat.

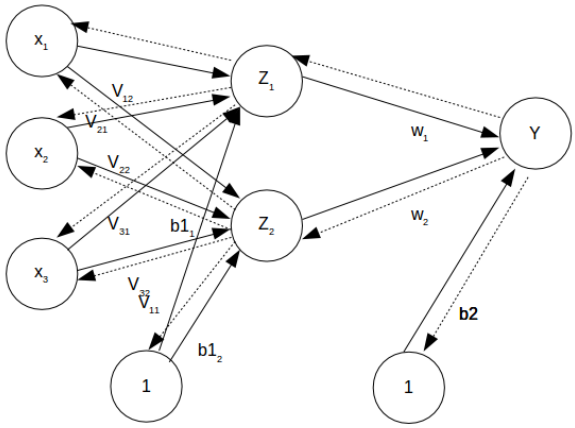
"*Deep Learning* memungkinkan komputasi model yang terdiri dari beberapa *processing layer* untuk mempelajari representasi data dengan macam-macam tingkat abstraksi. Metode ini telah memperbaiki *state of the art* dalam pengenalan suara *(speech recognition),* pengenalan objek visual *(visual object recognition),* deteksi objek *(object detection)* dan banyak penemuan lainnya seperti penemuan obat dan genomik. *Deep learning* menemukan struktur yang rumit dan sulit dalam kumpulan data yang sangat besar dengan menggunakan algoritma *backpropagation* untuk menunjukkan bagaimana sebuah mesin harus mengubah parameter internal yang digunakan untuk menghitung representasi pada setiap lapisan dari representasi pada lapisan sebelumnya (LeCun, Bengio & Hinton, 2015).

Beberapa algoritma yang menerapkan konsep *deep learning* antara lain *Convolutional Neural Network* (CNN) untuk melakukan klasifikasi gambar. *Deep Belief Network- Deep Neural Network* (DBN-DNN) untuk pengenalan suara, *Recurrent Neural Network* (RNN) untuk menerjemahkan bahasa. *Query-oriented* untuk meringkas multi dokumen. *Conditional Restricted Boltzman Machine*

(CRBM) untuk melakukan prediksi *Drag-Targer Interaction* (DTI) dan *Deep Belief Network* (DBN) untuk memprediksi data sesuai dengan waktu.

3.6 Pembelajaran *Backpropagation*

Menurut Kusumadewi (2014) *"Backpropagation* merupakan algoritma pembelajaran yang terawasi atau *supervised learning* dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan *neuron-neuron* yang ada pada lapisan tersembunyi. Pada algoritma ini, *error* yang digunakan adalah *error output* untuk mengubah nilai bobot-bobotnya dalam arah mundur *(backward)*. Jika ingin mendapatkan *error* ini, maka tahap perambatan maju *(forward propagation)* harus dikerjakan terlebih dahulu. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan menggunakan fungsi aktivasi yang dapat diturunkan, seperti fungsi *sigmoid*".



Gambar 3.9 Arsitektur jaringan *backpropagation*

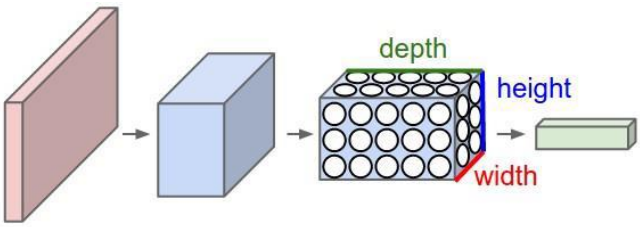
Gambar 3.9 di atas merupakan arsitektur jaringan *backpropagation.* Pada gambar tersebut terlihat bahwa *neural network* terdiri dari tiga unit *neuron* pada lapisan *input* (X_1, X_2 dan X_3), dua *neuron* pada *hidden layar* (Z_1 dan Z_2) dan 1 unit *neuron* pada lapisan *output* (Y). Bobot yang *menghubungkan* lapisan *input* dengan *neuron* pertama pada *hidden layar* adalah V_{11}, V_{21} dan V_{31}. Bobot bias yang menuju *neuron* pertama dan kedua pada *hidden layer* ditunjukkan dengan b_{11} dan b_{12}. Bobot yang mengubungkan Z_1 dan Z_2 dengan *neuron* pada lapisan *output* adalah W_1 dan W_2. Bobot bias b_2 menghubungkan *hidden layer* dengan lapisan *output*.

3.7 *Convolutional Neural Network*

Convolutional network atau yang dikenal juga dengan nama *convolutional neural network* (CNN) adalah tipe khusus dari neural network untuk memproses data yang mempunyai topologi jala atau *grid-like topology*. Pemberian nama *convolutional neural network* mengindikasikan bahwa jaringan tersebut menggunakan operasi matematika yang disebut konvolusi. Konvolusi sendiri adalah sebuah operasi linear. Jadi *Convolutional Network* adalah *neural network* yang menggunakan konvolusi minimal satu pada lapisannya (LeCun, Bengio & Hinton, 2015). CNN digunakan untuk melakukan klasifikasi data yang berlabel dengan menggunakan metode *supervised learning* yang cara kerjanya adalah terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini yaitu mengelompokkan suatu data ke data yang sudah ada.

CNN sering digunakan untuk mengenali benda atas pemandangan dan melakukan deteksi serta melakukan segmentasi objek. CNN belajar langsung dari data citra, sehingga dapat menghilangkan ekstraksi ciri dengan cara manual. Penelitian awal yang menjadi dasar penemuan ini yaitu pertama kali dilakukan oleh Hubel dan Wiesel yang melakukan penelitian *visual cortex* pada indera penglihatan kucing. *Visual cortex* pada hewan sangat *powerful* kemampuannya dalam sistem permrosesan visual yang pernah ada. Sehingga, banyak penelitian yang terinspirasi oleh cara kerjanya dan menghasilkan banyak model-model baru yang beberapa diantaranya yaitu, *Neocognition*, HMAX, LeNet-5 dan AlexNet.

CNN juga merupakan syaraf yang dikhusukan untuk memproses data yang memiliki struktur kotak. Sebagai contoh yaitu berupa data citra dua dimensi. Nama konvolusi merupakan operasi dari aljabar linear yang mengalikan matriks dari *filter* pada citra yang akan diproses. Proses ini disebut dengan lapisan konvolusi dan merupakan salah satu jenis dari banyak lapisan yang bisa dimiliki dalam suatu jaringan. Meskipun begitu, lapisan konvolusi ini merupakan lapisan utama yang paling penting digunakan. Jenis lapisan yang lain yang biasa digunakan adalah *pooling layer*, yakni lapisan yang digunakan untuk mengambil suatu nilai maksimal atau nilai rata-rata dari bagian-bagian lapisan piksel pada citra. Berikut merupakan arsitektur *convolution net:*



Gambar 3.10 Struktur dari CNN untuk citra

Berdasarkan **gambar 3.10** di atas, dapat diketahui bahwa setiap lapisan *input* yang dimasukkan mempunyai susunan neuron 3 dimensi, yaitu lebar, tinggi dan ke dalaman). Lebar dan tinggi merupakan ukuran lapisan, sedangkan ke dalaman adalah jumlah lapisan. Setiap besaran yang didapat terhitung dari hasil fitrasi dari lapisan sebelumnya dan banyaknya *filter* yang digunakan. Model jaringan seperti ini sudah terbukti efektif dalam menangani permasalahan klasifikasi citra. CNN mampu memiliki puluhan hingga ratusan lapisan yang masing-masing lapisan mempelajari deteksi berbagai gambar. Pengolahan citra diterapkan pada setiap citra latih pada resolusi yang berbeda dan *output* dari masing-masing data gambar yang diolah dan digunakan sebagai sebagai *input* ke lapisan berikutnya. Pengolahan citra dapat dimulai sebagai fitur yang sederhana, seperti ukuran kecerahan dan tepi atau meningkatkan kekompleksan pada fitur secara unik untuk menentukan objek sesuai ketebalan lapisan (MathWorks, 2018). Secara umum tipe lapisan CNN ada dua, yaitu:

1. Lapisan Ekstraksi Fitur (*Feature Extraction Layer)*

   Lapisan ini terletak di awal arsitektur. Neuron yang ada di dalam lapisan ini terhubung pada daerah lokal dari lapisan sebelumnya. *Convolutional layer* merupakan lapisan jenis pertama dan *pooling layer* merupakan lapisan kedua. Lapisan ini menerima *input* citra secara langsung kemudian memprosesnya sampai menghasilkan *output* berupa vektor untuk diolah di lapisan berikutnya.

2. Lapisan Klasifikasi *(Classfication Layer)*

   Lapisan ini tersusun atas beberapa lapisan dan setiap lapisan tersusun atas neuron yang terkoneksi secara penuh *(fully connected)* dengan lapisan lainnya. Lapisan ini menerima *input* dari hasil keluaran lapisan ekstrasi fitur gambar berupa vektor kemudian ditransformasikan seperti *Multi Neural*

Networks dengan tambahan beberapa *hidden layer*. Hasil keluaran berupa akurasi kelas untuk klasifikasi. Berikut adalah tingkatan nilai akurasi:

a. Nilai akurasi sebesar 0,90-1,00 berarti *excellent classification*
b. Nilai akurasi sebesar 0,80-0,90 berarti *good classification*
c. Nilai akurasi sebesar 0,70-0,80 berarti *fair classification*
d. Nilai akurasi sebesar 0,60-0,70 berarti *poor classification*
e. Nilai akurasi sebesar 0,50-0,60 berarti *failure*

3.7.1 Lapisan Konvolusi

Lapisan konvolusi terdiri dari tiga operasi utama yaitu operasi konvolusi, fungsi aktivasi dan *pooling*. Berikut akan dijelaskan mengenai tiga operasi utama tersebut:

Operasi Konvolusi

Operasi konvolusi yang dikenakan pada fungsi x(t) dengan bobot (atau sering disebut *kernel*) w(t), $x*w$, didefinisikan secara matematis pada persamaan (3). Pada persamaan tersebut, *s(t)* adalah fungsi hasil operasi konvolusi, t adalah variabel dari fungsi, dan a adalah konstan.

$$S(t) = (x * w)(t) = \sum_{a=\infty}^{\infty} x(a) * w(t-a) \quad (3)$$

Dalam aplikasi *machine learning*, *input* x adalah berupa *array* banyak dimensi yang memuat data. Sedangkan bobot w adalah *array* multi dimensi yang merupakan parameter yang bisa dipelajari. Karena *input* dan bobot harus disimpan sebelum pemrosesan, maka nilai *input* dan bobot dianggap nol di semua komponen kecuali komponen yang disimpan. Maka penjumlahan yang didapatkan adalah bukan penjumlahan tak berhingga tetapi sebanyak elemen a atau banyak *array input*.

Pada pengolahan citra digital, konvolusi dipahami dengan menggerakkan suatu *kernel* K berukuran mxn pada citra I yang berukuran ixj, dan kemudian mengambil jumlah dari perkalian nilai citra dan *kernel* tersebut. Istilah konvolusi hampir sama dengan istilah korelasi, bedanya pada saat konvolusi, *kernel* yang digunakan terlebih dahulu di balik 180^o. Pada aplikasi *machine learning*, kedua istilah tersebut dianggap sama, sehingga ketika konvolusi dilakukan, *kernel* boleh

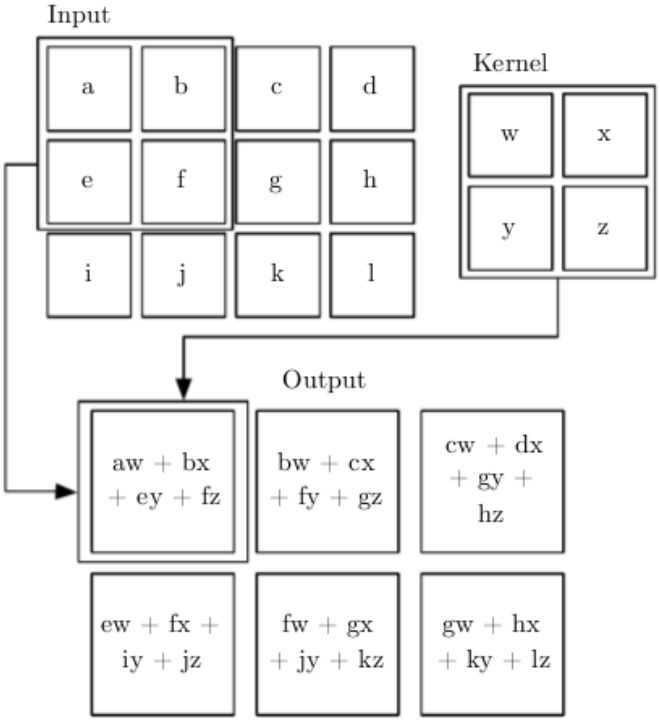
dibalik dulu ataupun tidak (Gonzales, Rafael C., Richard E. Woods dan Steven L. Eddins, 2008). Secara formal, konvolusi pada citra berukuran sxt, I(sxt), dengan *kernel* berukuran mxn, K(mxn), bisa dinyatakan dalam persamaan (4) sementara korelasi dinyatakan dengan persamaan (5).

$$s[i,j] = (I * K)[i,j] = \sum_m \sum_n I\,[i-m, j-n]\,K[m,n] \quad (4)$$

$$s[i,j] = (I * K)[i,j] = \sum_m \sum_n I\,[i+m, j+n]\,K[m,n] \quad (5)$$

Pada kedua persamaan tersebut, nilai (m,n) berpusat di tengah nilai *kernel* (0,0). Sementara koordinat citra (i,j) bernilai (0,0) pada sisi kiri atas citra. Ukuran citra hasil konvolusi berkurang dibandingkan dengan citra awal dan dapat dinyatakan dengan persamaan (6). Dalam hal ini jika citra dengan ukuran 32x32 dikenai konvolusi dengan ukuran *kernel* 3x3 maka ukuran akhir menjadi 32-3+1 x 32-3+1 = 30x30.

$$\text{Ukuran}_{hasilKonvolusi} = \text{Ukuran}_{\text{awal}} - filtersize + 1 \quad (6)$$

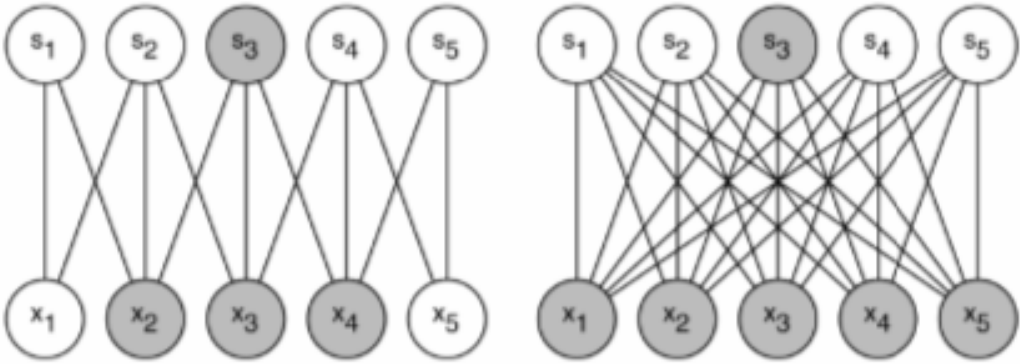


Gambar 3.11 Proses konvolusi pada *input array*

Ilustrasi proses konvolusi pada citra, yang merupakan *array* dua dimensi I, dengan bobot K (dua dimensi) ditunjukkan pada **gambar 3.11** di atas. Pada

gambar tersebut, citra berukuran 4x3 dikonvolusi dengan menggunakan *kernel* berukuran 2x2. Citra yang dihasilkan adalah berukuran 3x2. Elemen pertama pada citra hasil konvolusi adalah merupakan jumlah dari perkalian bobot *kernel* dengan nilai citra yang bersangkutan. Dalam penerapannya, tambahan *zero padding* bisa ditambahkan pada sisi citra input untuk menjaga informasi tepi. Proses konvolusi mengusung ide penting yang bisa meningkatkan hasil dari *machine learning system* yaitu: *sparse interactions* dan *parameter sharing***.** Selain itu, proses konvolusi juga menyediakan cara untuk berinteraksi dengan input dalam ukuran yang berbeda-beda.

Konsep *sparse interactions* ditunjukkan pada **gambar 3.12**. Pada gambar tersebut ditunjukkan hubungan *neural network* yang terdiri dari satu lapisan *input* dan lapisan *output*. Hubungan yang terjadi pada umumnya yang menggunakan perkalian matriks untuk mendeskripsikan interaksi antara lapisan *input* dan lapisan *output*. Dalam hal ini, setiap *output* berinteraksi dengan semua *input*. *Convolutional network* dalam hal ini ditunjukkan pada gambar di sisi kiri yang mempunyai interaksi yang jarang. Hal ini dilakukan dengan menggunakan ukuran *kernel* yang lebih kecil dibanding dengan *input*.



Gambar 3.12 Interaksi proses konvolusi dan *neural network*

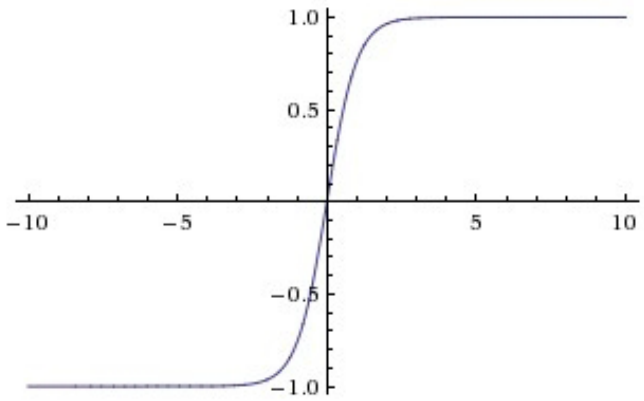
Sementara itu, parameter *sharing* adalah menggunakan parameter yang sama untuk lebih dari satu fungsi dalam sebuah model. Dalam *neural networks* yang biasa, masing-masing element dari matriks bobot digunakan tepat satu kali ketika menghitung *output*. Masing-masing bobot tersebut dikalikan dengan salah

satu elemen dari *input* dan tidak digunakan lagi dalam perhitungan selanjutnya. Dalam *convolutional network*, masing-masing anggota dari *kernel* digunakan pada setiap posisi dari *input*. Hal ini berarti pembelajaran parameter tidak dilakukan secara terpisah untuk setiap lokasi, akan tetapi pembelajaran hanya dilakukan ntuk satu set parameter untuk semua lokasi. Oleh karena itu, proses konvolusi menjadi lebih efisien dari perkalian matriks dari segi kebutuhan memori dan efisiensi statistik.

Fungsi Aktivasi Tanh

Elemen kedua dari lapisan konvolusi adalah fungsi aktivasi. Pada prose ini, fungsi aktivasi Tanh akan diaktifkan pada hasil konvolusi. Cara kerja fungsi ini adalah mengubah nilai *real* menjadi nilai antara -1 dan 1 seperti pada **gambar 3.12**, nilai yang sangat negatif dirubah menjadi nilai -1 sedangkan nilai yang sangat positif dirubah menjadi nilai 1. 0 adalah pusat nilai dari fungsi ini. Berikut adalah persamaan untuk fungsi *tanh:*

$$\tanh(a) = \frac{e^{a}-e^{-a}}{e^{a}+e^{-a}} \tag{7}$$

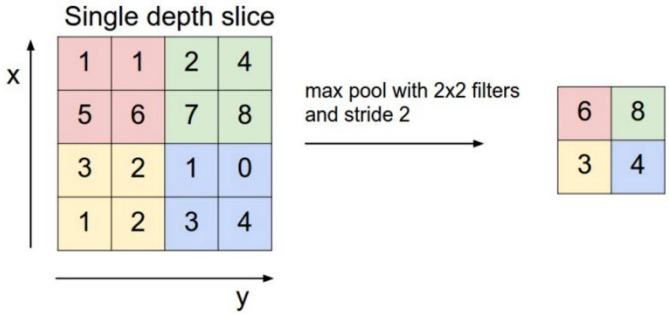


Gambar 3.13 Fungsi *tanh*

Operasi *Pooling*

Pooling atau *subsampling* adalah lapisan fungsi untuk *feature maps* sebagai masukan dan mengolahnya dengan berbagai operasi statistik berdasarkan nilai piksel terdekat. Pada model CNN, lapisan *pooling* biasanya disisipkan secara teratur setelah beberapa lapisan konvolusi. *Pooling layer* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam susunan arsitektur model CNN dapat secara progresif mengurangi ukuran volume *output* pada *feature*

maps, sehingga dapat mengurangi jumlah parameter dan untuk mengendalikan *overfitting.* Hal penting dalam pembuatan model CNN adalah dengan memilih banyak jenis *pooling layer* yang dalam hal ini dapat menguntungkan kinerja model. *Pooling layer* bekerja disetiap susunan *feature maps* dan mengurangi ukurannya. Bentuk *pooling layer* yang paling umum adalah dengan menggunakan *filter* berukuran 2x2 yang diaplikasikan dengan langkah sebanyak 2 dan kemudian beroperasi pada setiap irisan dan *input.* Bentuk seperti ini akan mengurangi *feature maps* hingga 75 dari ukuran aslinya.



Gambar 3.14 Proses *max-pooling*

Lapisan pada *pooling* akan berjalan pada setiap irisan ke dalaman volume *input* secara bergantian. Pada **gambar 3.14** di atas lapisan *pooling* menggunakan salah satu operasi maksimal *(max-pooling)* yang merupakan operasi yang paling umum. Penggunaan *pooling layer* pada CNN bertujuan untuk mereduksi ukuran citra sehingga dapat dengan mudah digantikan dengan sebuah *convolution layer* dengan *stride* yang sama dengan *pooling layer* yang bersangkutan.

3.7.2 *Fully-Connected Layer*

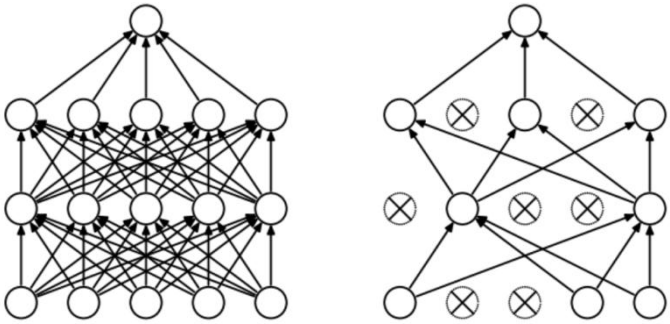
Fully-connected layer merupakan lapisan dimana seluruh neuron aktivasi dari lapisan sebelumnya terhubung dengan neuron pada lapisan selanjutnya seperti halnya jaringan syaraf tiruan biasa. Setiap aktivasi dari lapisan sebelumnya perlu diubah menjadi data satu dimensi sebelum dapat dihubungkan ke semua neuron di lapisan *fully connected. Fully-connected layer* biasanya digunakan pada metode *Multilayer Perceptorn* dan gunanya untuk mengolah data sehingga dapat diklasifikasikan.

Perbedaan antara *fully-connected layer* dengan lapisan konvolusi biasa yaitu neuron di lapisan konvolusi hanya terhubung ke daerah tertentu pada *input*,

sementara *fully-connected layer* memiliki neuron yang secara keseluruhan akan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan *dot product*, sehingga fungsinya tidak begitu berbeda.

3.7.3 *Dropout Regularization*

Dropout merupakan teknik regularisasi jaringan syaraf tiruan dimana beberapa akan dipilih secara *random* dan tidak dipakai selama data latih. Neuron-neuron ini dibuang juga secara *random.* Hal ini berarti bahwa kontribusi neuron yang dibuang akan dihentikan sementara jaringan dan bobot baru juga tidak ditetapkan pada neuron saat melakukan *backpropagation.* Teknik ini sangat mudah diterapkan pada model CNN sehingga akan berdampak pada performa model dalam melatih serta mengurangi *overfitting.*



Gambar 3.15 Pengaplikasian *dropout regularization*

3.7.4 *Softmax Classifier*

Softmax classifier adalah bentuk lain dari algoritma regresi logistik yang dapat digunakan untuk melakukan klasifikasi lebih dari dua kelas. Standar dari klasifikasi yang biasanya dilakukan oleh algoritma dari regresi logistik adalah tugas untuk klasifikasi kelas biner. *Softmax* mempunyai bentuk persamaan sebagai berikut:

$$Fj(z) = \frac{e^{z}j}{\sum_k e^{2}k} \tag{8}$$

Pada persamaan (8) di atas, untuk notasi f_j menunjukkan hasil fungsi untuk setiap elemen ke-j pada vektor *output* kelas. Argumen z merupakan hipotesis yang diberikan oleh model pelatihan supaya dapat diklasifikasikan oleh fungsi *softmax. Softmax* juga memberikan hasil yang lebih intuitif dan memiliki hasil interpretasi probabilistik yang lebih baik dibandingkan dengan algoritma klasifikasi lainnya.

Softmax memungkinkan peneliti untuk menghitung nilai probabilitas untuk semua label. Hasil dari label yang ada, akan diambil sebuah vektor nilai yang mempunyai nilai rill dan merubahnya menjadi vektor dengan nilai antara nol dan satu. Jika hasil dijumlah maka akan bernilai satu.

3.8 MXNet

MXNet adalah perpustakaan pembelajaran *machine learning* yang digunakan untuk mempermudah pengembangan algoritma *machine learning*, terutama untuk *deep neural networks.* Keunggulan MXNet adalah mendukung komputasi terdistribusi. Komputasi pada MXNet menggabungkan antara ekspresi simbolis dengan tensor imperatif. Komputasi ini akan memberikan diferensiasi otomatis pada gradien. MXNet merupakan perhitungan memori yang efisien dan berjalan pada sistem heterogen, dari perangkat *mobile* ke klasifikasi GPU. MXNet dapat dijalankan pada GPU ataupun CPU. Keunggulan lainnya MXNet dapat membangun arsitektur jaringan dari awal yang dapat di kustom sedemikian rupa untuk menghasilkan akurasi yang baik. Serta pelatihan data lebih cepat dan tidak membutuhkan banyak *space memory.*

BAB IV
METODOLOGI PENELITIAN

4.1 Populasi dan Sampel Penelitian

Populasi yang digunakan dalam penelitian ini adalah semua citra uang kertas rupiah tahun emisi 2017 pecahan Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00. Sedangkan, untuk sampel yang digunakan dalam penelitian ini adalah 700 citra uang kertas rupiah tahun emisi 2017 dengan setiap kategori (pecahan Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00) terdiri dari 100 citra.

4.2 Variabel dan Definisi Operasional

Definisi operasional variabel penelitian merupakan penjelasan dari masing-masing variabel yang digunakan dalam penelitian terhadap indikator-indikator yang membentuknya. Definisi operasional penelitian ini dapat dilihat pada **tabel 4.1** berikut:

Tabel 4.1 Definisi operasional variabel

Citra	Variabel	Definisi Operasional Variabel
	Rp 1.000,00	Citra uang kertas pecahan Rp 1.000,00 dari sisi depan dan sisi belakang dengan *background* putih.
	Rp 2.000,00	Citra uang kertas pecahan Rp 2.000,00 dari sisi depan dan sisi belakang dengan *background* putih.
	Rp 5.000,00	Citra uang kertas pecahan Rp 5.000,00 dari sisi depan dan sisi belakang dengan *background* putih.

	Rp 10.000,00	Citra uang kertas pecahan Rp 10.000,00 dari sisi depan dan sisi belakang dengan *background* putih.
	Rp 20.000,00	Citra uang kertas pecahan Rp 20.000,00 dari sisi depan dan sisi belakang dengan *background* putih.
	Rp 50.000,00	Citra uang kertas pecahan Rp 50.000,00 dari sisi depan dan sisi belakang dengan *background* putih.
	Rp 100.000,00	Citra uang kertas pecahan Rp 100.000,00 dari sisi depan dan sisi belakang dengan *background* putih.

4.3 Sumber Data

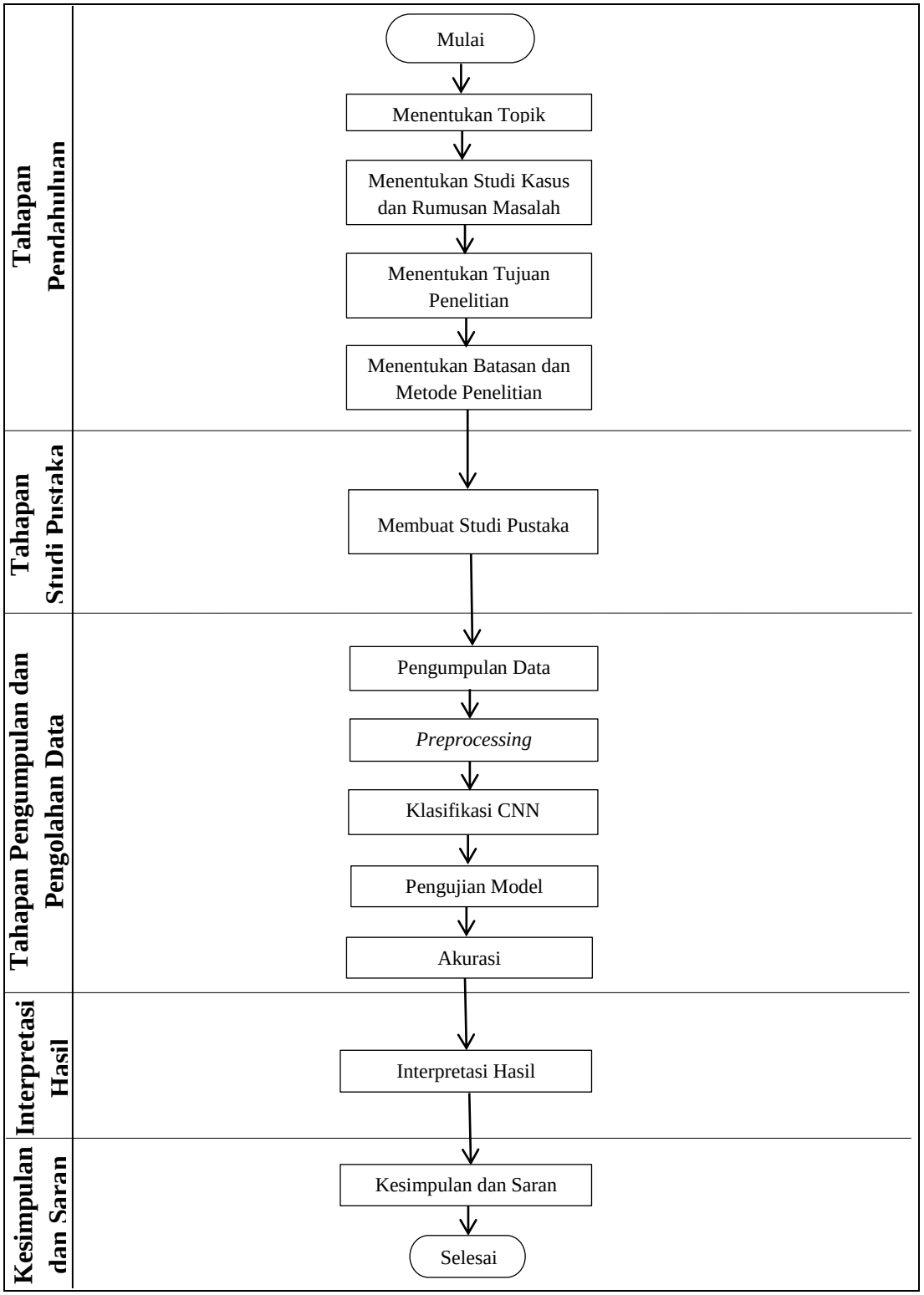
Penelitian ini menggunakan data primer yang didapatkan dengan cara mengambil citra uang kertas tahun emisi 2017 secara langsung dengan menggunakan kamera iphone 6s.

4.4 Metode Analisis Data

Metode analisis data yang digunakan dalam penelitian ini adalah metode *Deep Learning* dengan algoritma *Convolutional Neural Network* (CNN) untuk klasifikasi citra nominal uang kertas rupiah tahun emisi 2017. *Software* yang digunakan dalam penelitian ini adalah RStudio 1.1.383 dengan *framwork* MXNet.

4.5 Tahapan Penelitian

Tahapan yang dilakukan pada penelitian ini dapat digambarkan dalam *flowchart* berikut



Gambar 4.1 *Flow chart*

BAB V
PEMBAHASAN

5.1 Pengumpulan Data Citra

Tahapan pertama yang dilakukan dalam penelitian ini adalah adalah mengumpulkan data citra. Pada penelitian ini, data citra diambil dengan menggunakan kamera utama *Iphone* 6s dengan resolusi kamera 12 megapiksel. Ukuran gambar yang dihasilkan dari kamera tersebut adalah ± 2,5 megapiksel. Data citra diambil diluar ruangan dengan pencahayaan yang cukup. Berikut adalah gambar *iphone* 6s yang digunakan:



Gambar 5.1 *Iphone* 6s

Cara mengambil data citra adalah dengan meletakkan uang kertas di atas lantai yang berwarna putih. Hal ini dilakukan agar *background* dari setiap data citra sama, yaitu putih. Selain itu, kamera hanya akan fokus ke objek uang kertas yang akan diambil saja, agar kualitas citra yang didapatkan baik dan sama pada setiap pecahannya. Sisi yang diambil adalah sisi bagian depan yang bergambar pahlawan Indonesia dan sisi bagian belakang yang bergambar tarian tradisional dari beberapa provinsi di Indonesia. Dalam penelitian ini, uang kertas akan diklasifikasikan menjadi 7 kategori yaitu pecahan uang Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00. Data penelitian yang diambil berjumlah 700 citra dengan jumlah masing-masing kategori sebanyak 100 citra. Kondisi uang kertas yang digunakan pada penelitian ini adalah uang kertas yang ada bekas lipatan. Penggunaan uang kertas yang sudah lusuh (banyak bekas lipatan atau lecek) merupakan salah satu kelebihan

sistem dibandingkan dengan sistem pada mesin penjualan otomatis *(vending machine)* karena sistem pada *vending machine* tidak dapat mendeteksi uang kertas dengan banyak bekas lipatan. Berikut adalah contoh gambar pecahan uang kertas yang digunakan:



Gambar 5.2 Pecahan uang kertas sisi bagian depan



Gambar 5.3 Pecahan uang kertas sisi bagian belakang

5.2 *Preprocessing* Citra

Setelah pengumpulan citra, tahap kedua adalah melakukan *preprocesing*. *Preprocessing* dilakukan untuk menyiapkan citra agar dapat diproses pada tahap klasifikasi CNN. Pada tahap ini, citra akan di*resize* dan warna citra akan diubah menjadi *grayscale.* Hal tersebut perlu dilakukan karena laptop yang dipakai untuk penelitian ini tidak dapat memproses citra dengan ukuran aslinya dan berwarna *(original)* dengan alasan performa laptop. Oleh karena itu, perlu dilakukan tahap *preprocessing* agar citra-citra tersebut dapat dianalisis dengan metode CNN.

Citra uang kertas akan di*resize* menjadi ukuran 32x32 *pixel.* Ukuran citra untuk masukan CNN yang dianjurkan adalah nilai pangkat dua (2^n). Dengan alasan komputasi dan performa laptop maka ukuran yang dipilih adalah 32x32. Citra yang berukuran 32x32 *pixel* ini diproses dan diubah ke dalam bentuk matriks sebagai masukan bagi proses klasifikasi dengan CNN.

Selanjutnya adalah mengubah warna citra menjadi *grayscale.* Proses pertama yang dilakukan untuk mengubah warna citra adalah dengan mengambil citra warna sebagai masukan. Citra warna, tinggi dan lebar untuk kebutuhan pembuatan citra baru. Kemudian dilakukan pemisahan terhadap nilai dari masing-masing komponen *red, green* dan *blue* dari citra warna tersebut. Selanjutnya, satu citra baru dibuat untuk menampung citra hasil perubahan model warna. *Output* dari tahap ini adalah citra ukuran 32x32 dengan warna *grayscale*. Berikut adalah *script* yang digunakan pada tahap *preprocessing:*

```
library(EBImage)
setwd("E://1 seribu")
# Set label
labelseribu <- 0
# Set up df
df <- data.frame()
# List images in path
seribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(seribu))
{

  uangseribu <- readImage(seribu[i])
  uangseribu_resized <- resize(uangseribu, w=w, h=h)
  ## Set to grayscale
  uangseribu_gray <- channel(uangseribu_resized, "gray")
  ## Get the image as a matrix
  uangseribu_matrix <- uangseribu_gray@.Data
  ## Coerce to a vector
  uangseribu_vector <- as.vector(t(uangseribu_matrix))
  seribuku=c(labelseribu,uangseribu_vector)
  df<-rbind(df,seribuku)
  write.csv(df, "seribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uangseribu_gray)
```

Gambar 5.4 *Script* tahap *preprocessing*

Sebelum memulai *preprocessing* citra, aktifkan *package* EBImage terlebih dahulu dengan menggunakan *script library* seperti pada *line* 1. *Package* EBImage digunakan untuk membaca dan menampilkan data citra. Kemudian labelkan citra uang seribu dengan menggunakan *script* pada *line* 4. Pada penelitian ini, uang seribu dilabelkan dengan angka 0, uang dua ribu dilabelkan dengan angka 1, uang lima ribu dilabelkan dengan angka 2 dan seterusnya. Kemudian masuk ke tahap *preprocessing* citra. *Script* pada line 10 dan 12 menunjukkan ukuran citra disimbolkan dengan *height* atau h dan *widht* atau w dalam dua dimensi (h x w). *Script* pada *line* 15 adalah *script* yang digunakan untuk merubah ukuran citra. *Script* pada *line* 21 adalah *script* yang digunakan untuk mengubah citra menjadi *grayscale.* Kemudian citra *grayscale* dengan ukuran 32x32 tersebut akan diubah ke dalam bentuk matriks dan diubah lagi menjadi bentuk vektor, *script* yang digunakan terdapat pada *line* 23 dan 25. Setelah itu, label uang seribu dan nilai vektor disimpan ke dalam *format* .csv *(comma delimited value)* dengan nama `seribuku.csv` dengan *script* pada *line* 28. Di dalam *file* `seribuku.csv` terdapat satu kolom label 0 sebanyak 100 baris dan kolom piksel sebanyak 1024 untuk masing-masing citra. Jumlah 100 diperoleh dari jumlah citra pada kategori seribu dan 1024 diperoleh dari 32x32. Selanjutnya *file* tersebut akan digunakan untuk membangun data *set*.

5.3 Pembuatan Data *Set*

Tahapan ketiga setelah *preprocessing* adalah membuat data *set.* Data *set* merupakan representasi matriks dari citra-citra beserta label citra. Fungsi pembuatan data *set* adalah agar *format* masukan yang diterima oleh proses pelatihan konvolusi dengan menggunakan *library* MXNet. Pada penelitian ini, digunakan 2 data *set* yang terpisah, yaitu data *train* dan data *test.* Data *train* digunakan untuk melakukan *training* pada model dan data *test* digunakan untuk validasi dari data *training.* Jumlah data citra yang digunakan sebanyak 700 citra dengan masing-masing kategori terdiri dari 100 citra. Pada *script* **gambar 5.6**, peneliti menggunakan 80% data *train* dan 20% data *test,* maka data yang

di*training* sebanyak 560 data dan 140 data *testing*. Berikut adalah *script* yang digunakan untuk membuat data *set:*

```
seribucsv<- read.csv("E://1 seribu/seribuku.csv")
duaribucsv<- read.csv("E://2 dua ribu/duaribuku.csv")
limaribucsv<- read.csv("E://3 lima ribu/limaribuku.csv")
sepuluhribucsv<- read.csv("E://4 sepuluh ribu/sepuluhribuku.csv")
duapuluhribucsv<- read.csv("E://5 dua puluh ribu/duapuluhribuku.csv")
limapuluhribucsv<- read.csv("E://6 lima puluh ribu/limapuluhribuku.csv")
seratusribucsv<- read.csv("E://7 seratus ribu/seratusribuku.csv")
dim(seribucsv)
```

Gambar 5.5 Memanggil data *set*

Script di atas digunakan untuk memanggil data *set* yang telah disimpan dalam bentuk csv pada masing-masing kategori uang kertas. Kemudian untuk menampilkan dimensi data dapat menggunakan perintah `dim`.

```
library(caret)
complete_set <- rbind(seribucsv,duaribucsv,limaribucsv,
                      sepuluhribucsv,duapuluhribucsv,limapuluhribucsv,
                      seratusribucsv)
training_index <- createDataPartition(complete_set$label, p = .8, times = 1)
training_index <- unlist(training_index)
train_set <- complete_set[training_index,]
dim(train_set)
test_set <- complete_set[-training_index,]
dim(test_set)
train_data <- data.matrix(train_set)
train_x <- t(train_data[, -1])
dim(train_x)
train_y <- train_data[,1]
dim(train_y)
train_array <- train_x
dim(train_array)
dim(train_array) <- c(32, 32, 1, ncol(train_x))

test_data <- data.matrix(test_set)
test_x <- t(test_set[,-1])
dim(test_x)
test_y <- test_set[,1]
test_array <- test_x
dim(test_array)
dim(test_array) <- c(32, 32, 1, ncol(test_x))
```

Gambar 5.6 *Script* untuk membangun datasets

Selanjutnya aktifkan *library* caret untuk mempartisi dua data *set*, yaitu data *train* dan data *test.* Kemudian gabungkan semua data *set* yang telah dipanggil sebelumnya dengan menggunakan perintah pada *line* 243-245. *Script* di atas menunjukkan penggunaan data *train* sebanyak 80% dari total data terlihat pada *line* 242. *Script* pada *line* 243 digunakan untuk mengubah bentuk `train_index` menjadi bentuk vektor. *Script* pada *line* 244 digunakan agar data yang digunakan untuk *train set* adalah data dari `train_index`, sedangkan data untuk *test set* adalah sisa datanya. Data *train* dan data *test* terdiri dari *file* csv yang telah

digabungkan menjadi `complete_set`. *Script* pada *line* 248 dan 257 digunakan untuk mengubah `train_set` dan `test_set` ke dalam bentuk matriks. *Script* pada *line* 249 dan 260 adalah perintah yang digunakan untuk menghilangkan kolom label di dalam data *train* dan data *set.*

5.4 Proses Klasifikasi CNN

Tahapan yang keempat adalah klasifikasi CNN. Prinsip dasar dari tahap ini adalah dengan melakukan pelatihan pada CNN untuk mendapatkan model yang mampu memberikan akurasi terbaik. Proses pertama yang dilakukan adalah membaca data *set.* Dari data *set* tersebut diambil data *train* dan data *test.* Data *train* kemudian digunakan untuk melakukan pelatihan CNN. Proses pelatihan CNN diawali dengan inisialisasi parameter dan dilakukan berulang-ulang sesuai maksimum perulangan yang ditentukan. *Output* dari proses ini adalah model terbaik untuk melakukan klasifikasi. Akurasi model tersebut dapat ditentukan dengan melakukan validasi dengan menggunakan data uji. Cara kerja pembacaan data adalah dengan membuka data *set* yang telah dipersiapkan pada tahapan sebelumnya untuk digunakan pada saat pelatihan dan perhitungan akurasi. Setelah dilakukan pembacaan dan penyalinan, maka *file* yang berisi data *set* ditutup kembali. Kemudian data *file* diurai menjadi data *pixel* dan data label. Data-data tersebut dikembalikan sebagai masukan bagi proses selanjutnya.

Parameter yang digunakan dalam penelitian ini adalah optimasi, jumlah *epoch,* banyaknya lapisan konvolusi, jumlah *filter* pada masing-masing lapisan, ukuran kernel dan jumlah neuron pada lapisan tersembunyi yang digunakan. Proses pelatihan model CNN sama dengan cara kerja proses pelatihan model jaringan syaraf tiruan lainnya, yaitu dengan cara melakukan *forward propagation* dan *backward propagation* sebanyak iterasi yang ditentukan. Pada penelitian ini, pembaharuan nilai bobot dilakukan dengan *optimizer adam.* Inisialisasi parameter dilakukan dengan menentukan jumlah *epoch,* optimasi dan ukuran kernel pada masing-masing proses dilapisan konvolusi.

Pelatihan model CNN menggunakan masukan citra warna dengan ukuran 32x32x1. Kemudian citra tersebut akan melalui proses konvolusi. Penelitian ini

menggunakan 2 lapisan konvolusi. Banyaknya lapisan konvolusi akan menentukan akurasi model klasifikasi. Proses konvolusi dilakukan terhadap kernel berukuran 1 dimensi. Proses konvolusi dilapisan kedua menggunakan masukan citra yang berasal dari proses konvolusi sebelumnya.

Selanjutnya dilakukan proses *pooling* dengan tujuan mengurangi matriks. Terdapat tiga macam metode *pooling* yaitu, *average pooling, max-pooling* dan *sum-pooling.* Dalam *average pooling,* nilai yang diambil adalah nilai rata-rata, sementara pada *max-pooling,* nilai yang diambil adalah nilai maksimal, dan *sum-pooling* menjumlahkan semua nilai. Metode yang digunakan pada penelitian ini adalah *max-pooling.* Alasan menggunakan *max-pooling* adalah jika nilai matriks pada lapisan *pooling* terdiri dari sedikit nilai besar dan banyak nilai kecil dan mengambil nilai besarnya saja maka akan mendapatkan informasi gambar lebih banyak sehingga akan menghasilkan akurasi yang lebih baik. Pada penelitian ini, juga dilakukan *trail and error* untuk membandingkan akurasi dari ketiga metode *pooling* tersebut karena pada beberapa kasus, *average pooling* atau *sum-pooling* akan menghasilkan akurasi yang lebih baik. *Output* dari proses konvolusi akan menjadi *input* untuk proses *max-pooling* sedangkan *output* dari proses *max-pooling* adalah nilai terbesar dari hasil konvolusi. Ukuran *kernel* dan *stride* pada proses *pooling* untuk *layer* pertama dan *layer* kedua sama, yaitu 2x2. Artinya setiap langkah berukuran 2x2 akan dilakukan *pooling* dan langkah akan bergeser dua ke kanan atau ke bawah. Fungsi aktivasi yang digunakan pada proses *pooling* adalah fungsi Tanh.

Proses selanjutnya berada di*fully-connected layer. Layer* ini merupakan *hidden layer* pada umumnya dijaringan syaraf tiruan yang berbentuk satu dimensi. *Input* pada *fully-connected layer* berasal dari proses yang ada dilapisan konvolusi sebelumnya. Masing-masing komponen matriks dari keluaran lapisan konvolusi dianggap sebagai satu nilai masukan bagi *fully-connected layer.* Ada dua *Fully-connected layer* yang digunakan pada penelitian ini. Pada proses ini, fungsi aktivasi yang digunakan adalah Tanh.

Proses terakhir pada tahapan klasifikasi CNN adalah *output layer,* yang merupakan *layer* terakhir dan terdiri dari 7 neuron yang mewakili 7 pecahan uang

kertas yang ingin diklasifikasikan. *Layer* ini terhubung secara penuh dengan *layer* sebelumnya. *Layer* ini merupakan *softmax layer* sehingga jika nilai aktivasi pada masing-masing neuron ditotal akan bernilai 1. *Softmax* merupakan perintah yang digunakan untuk mengklasifikasikan objek pada penelitian ini. *Output* dari proses klasifikasi CNN adalah hasil klasifikasi dan nilai akurasi model. Berikut adalah *script* yang digunakan pada proses klasifikasi CNN:

```
278 library(mxnet)
279 mx_data <- mx.symbol.Variable('data')
280 conv_1 <- mx.symbol.Convolution(data = mx_data, kernel = c(3, 3), num_filter = 20)
281 tanh_1 <- mx.symbol.Activation(data = conv_1, act_type = "tanh")
282 pool_1 <- mx.symbol.Pooling(data = tanh_1, pool_type = "max", kernel = c(2, 2), stride = c(2,2))
283 conv_2 <- mx.symbol.Convolution(data = pool_1, kernel = c(2,2), num_filter = 20)
284 tanh_2 <- mx.symbol.Activation(data = conv_2, act_type = "tanh")
285 pool_2 <- mx.symbol.Pooling(data = tanh_2, pool_type = "max", kernel = c(2, 2),stride = c(2, 2))
286 flat <- mx.symbol.Flatten(data = pool_2)
287 drop1 = mx.symbol.Dropout(data=flat, p=0.1)
288 fcl_1 <- mx.symbol.FullyConnected(data = drop1, num_hidden = 10)
289 tanh_3 <- mx.symbol.Activation(data = fcl_1, act_type = "tanh")
290 drop1 = mx.symbol.Dropout(tanh_3, p=0.1)
291 fcl_2 <- mx.symbol.FullyConnected(data = drop1, num_hidden = 7)
292 NN_model <- mx.symbol.SoftmaxOutput(data = fcl_2)
```

Gambar 5.7 *Script* untuk pelatihan CNN

Library yang digunakan untuk mengklasifikasikan objek adalah MXNet. Sama seperti sebelumnya, aktifkan terlebih dahulu *library* tersebut. Selanjutnya adalah proses konvolusi lapisan pertama. Hal pertama yang dilakukan adalah membuat sebuah mesin untuk menerima *input* dari citra yang telah diubah menjadi matriks. Pada penelitian ini ukuran kernel yang digunakan pada konvolusi *layer* pertama adalah 3x3 dengan jumlah *filter* 20. Fungsi aktivasi yang digunakan adalah Tanh. Metode yang digunakan pada proses *pooling layer* adalah *max-poling* dengan ukuran kernel 2x2 dan *stride* 2x2. Hasil dari proses *pooling* inilah yang nantinya akan menjadi nilai masukan pada konvolusi lapisan kedua. Pada konvolusi lapisan kedua, ukuran kernel diubah menjadi 2x2 dengan *filter* yang sama yaitu 20. Fungsi aktivasi dan proses *pooling* untuk konvolusi lapisan kedua masih sama seperti pada konvolusi lapisan pertama.

Selanjutnya adalah pembuatan *flatten* dari hasil *pooling* kedua pada lapisan kovolusi kedua pula. Kemudian data *flat* diregulasi pada proses *dropout* pertama dengan menggunakan probabilitas sebesar 10%. Data *dropout* tersebut akan dijadikan masukan bagi proses *fully-connected layer* pertama. Pada proses *fully-connected layer* pertama, neuron yang digunakan sebanyak 10 seperti pada *line* 288. Pada lapisan ini masih menggunakan fungsi aktivasi Tanh. Setelah itu,

hasil aktivasi fungsi Tanh diregulasi dan hasil dari regulasi tersebut akan dijadikan masukan bagi proses *fully-connected layer* kedua. Pada proses *fully-connected layer* pertama, neuron yang digunakan sebanyak 7 yang mewakili 7 pecahan uang kertas.

```
299 model <- mx.model.FeedForward.create(NN_model, X = train_array, y = train_y,
300                                      ctx = device,
301                                      num.round = 50,
302                                      array.batch.size = 100,
303                                      optimizer = "adam",
304                                      eval.metric = mx.metric.accuracy,
305                                      initializer = mx.init.Xavier(factor_type = "in", rnd_type = "gaussian",magnitude = 2),
306                                      array.layout = "auto")
307 
308 predict_probs <- predict(model, train_array)
309 predict_probs
310 predicted_labels <- max.col(t(predict_probs))-1
311 table(train_data[, 1], predicted_labels)
312 sum(diag(table(train_y, predicted_labels)))/560
313 
314 predict_probs <- predict(model, test_array)
315 predict_probs
316 predicted_labels <- max.col(t(predict_probs))-1
317 table(test_data[, 1], predicted_labels)
318 sum(diag(table(test_y, predicted_labels)))/140
```

Gambar 5.8 Pembuatan dan prediksi model CNN

Proses yang terakhir adalah membuat model CNN dan model tersebut akan diuji pada data *train* dan data *test*. Pada *script* di atas, digunakan 50 *epoch* dengan *batch size* sebesar 100. *Script* pada line 308 digunakan untuk memprediksi peluang antara model dan `train_array`. Jika ingin mengetahui apakah sistem yang sudah dibuat dan dilatih sebelumnya dapat mengklasifikasikan objek atau tidak untuk data *test*, gunakan perintah pada *script* 310. Jika ingin mengetahui apakah sistem yang sudah dibuat dan dilatih sebelumnya dapat mengklasifikasikan objek atau tidak untuk data *train*, gunakan perintah pada *script* 316. Akurasi pada data *test* dan data *train* dapat ditampilkan dengan menggunakan *script* pada *line* 312 dan 318.

5.5 Parameter Pengujian

Dalam penelitian ini akan dilakukan pengujian 3 parameter untuk mendapatkan akurasi terbaik pada klasifikasi CNN. Parameter yang akan diubah-ubah adalah ukuran kernel, jumlah *epoch* dan jumlah data. Pada pengujian yang pertama, parameter yang akan dibandingkan adalah ukuran kernel pada lapisan konvolusi pertama dan kedua. Ada 6 perbandingan ukuran kernel pada **tabel 5.1** sebagai berikut:

Tabel 5.1 Perbandingan ukuran kernel

No	Konvolusi	Ukuran Kernel
1	Conv_1	5x5
	Conv_2	5x5
2	Conv_1	5x5
	Conv_2	3x3
3	Conv_1	5x5
	Conv_2	2x2
4	Conv_1	3x3
	Conv_2	3x3
5	Conv_1	3x3
	Conv_2	2x2
6	Conv_1	2x2
	Conv_2	2x2

Pada pengujian yang kedua, parameter yang akan dibandingkan adalah jumlah *epoch.* Jumlah *epoch* yang akan digunakan adalah 30, 50 dan 100. Pengujian terakhir akan membandingkan jumlah data *train* dan data test. Kondisi yang pertama akan digunakan 90% data *train* dari total data atau 630 citra dan 10% data *test* dari total data atau 70 citra. Kondisi yang kedua akan digunakan 80% data *train* dari total data atau 560 citra dan 20% data *test* dari total data atau 140 citra. Kondisi yang ketiga akan digunakan 70% data *train* dari total data atau 490 citra dan 30% data *test* dari total data atau 210 citra. Parameter yang nilainya tetap di awal dan tidak berubah selama pengujian dapat dilihat pada **tabel 5.2**. Hal ini dilakukan agar tidak terlalu banyak parameter yang diujikan.

Tabel 5.2 Nilai parameter yang tidak berubah selama pengujian

Parameter	Nilai
Filter pada konvolusi 1 dan 2	20
Probabilitas *dropout*	0,1
Batch size	100

5.6 Hasil dan Pembahasan

Pada sub bab ini akan dijelaskan mengenai *output* dari *script* yang telah dijelaskan.

5.6.1 Hasil *Preprocessing* Citra

Output dari *preprocessing* citra adalah citra ukuran 32x32 dan citra *grayscale* dengan ukuran 32x32. Berikut adalah salah satu contoh *output* tersebut:



(a) (b) (c) (d)



(e) (f) (g)

Gambar 5.9 *Output* proses *preprocessing*

Gambar 5.9 di atas merupakan *output* dari proses *preprocessing*. Gambar (a) adalah citra Rp 1.000,00, gambar (b) adalah citra Rp 2.000,00, gambar (c) adalah citra Rp 5.000,00, gambar (d) adalah citra Rp 10.000,00, gambar (e) adalah citra Rp 20.000,00, gambar (f) adalah citra Rp 50.000,00 dan gambar (g) adalah citra Rp 100.000,00.

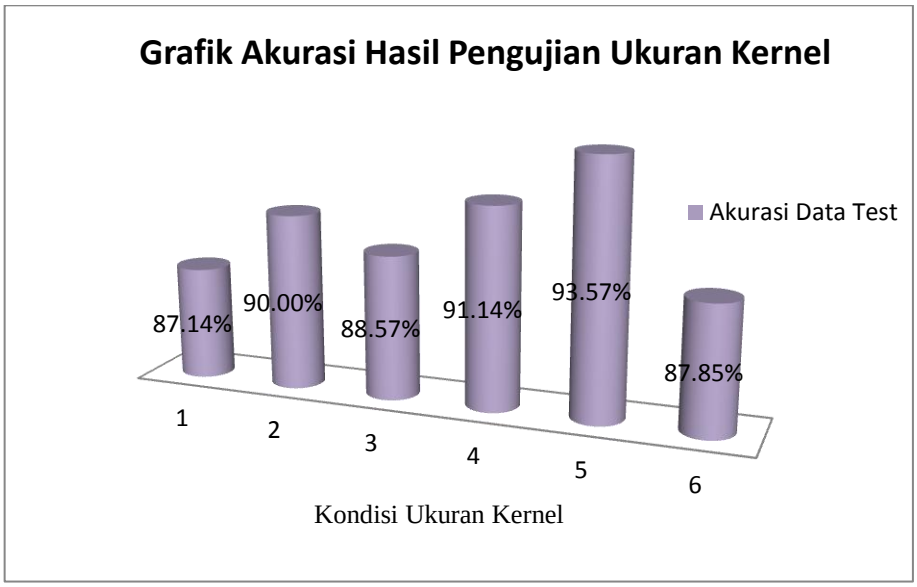
5.6.2 Hasil Pengujian Parameter

Telah dijelaskan pada sub bab sebelumnya ada 3 parameter yang akan dibandingkan nilai atau ukurannya untuk mendapatkan akurasi terbaik. Pengujian pertama adalah perbandingan akurasi dari ukuran kernel. Berikut adalah hasil dari pengujian tersebut:

Tabel 5.3 Hasil pengujian ukuran kernel

Kondisi	Konvolusi	Ukuran Kernel	Akurasi Data *Train*	Akurasi Data *Test*
1	Conv_1	5x5	100%	87,14%
	Conv_2	5x5		
2	Conv_1	5x5	100%	90,00%
	Conv_2	3x3		
3	Conv_1	5x5	100%	88,57%
	Conv_2	2x2		
4	Conv_1	3x3	100%	91,14%
	Conv_2	3x3		
5	Conv_1	3x3	100%	93,57%
	Conv_2	2x2		
6	Conv_1	2x2	100%	87,85%
	Conv_2	2x2		

Berdasarkan **tabel 5.3** di atas, dapat dilihat bahwa akurasi terbaik terdapat pada nomor 5, yaitu ukuran kernel 3x3 untuk lapisan konvolusi pertama dan 2x2 untuk lapisan konvolusi kedua. Berikut adalah grafik akurasi dari hasil pengujian ukuran kernel:



Gambar 5.10 Grafik akurasi pengujian ukuran kernel

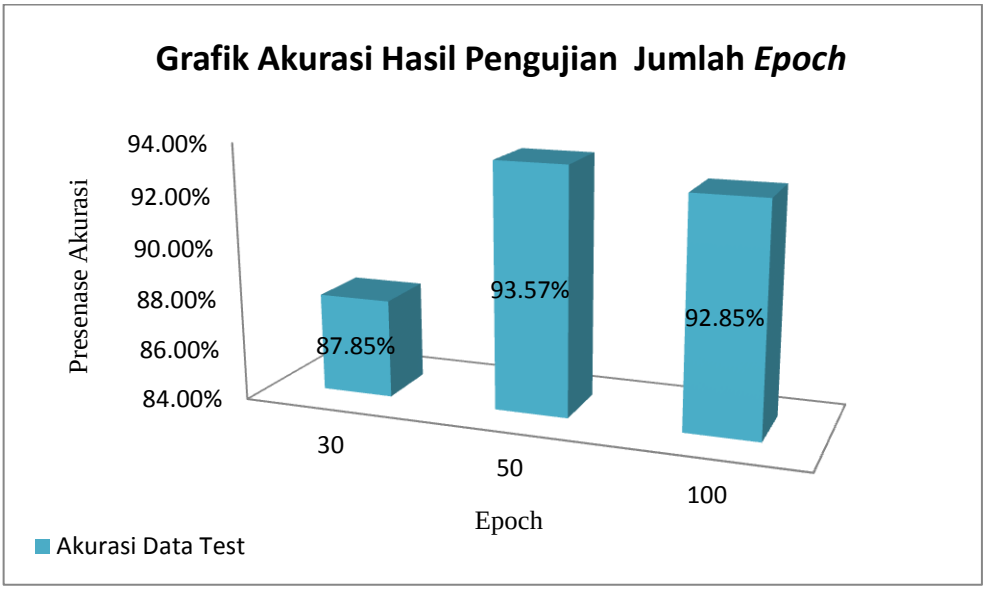
G**ambar 5.10** di atas, dapat diinterpretasikan bahwa ukuran kernel 3x3 pada konvolusi pertama dan 2x2 pada konvolusi kedua dapat mengklasifikasikan citra nominal uang kertas tahun emisi 2017 dengan menghasilkan akurasi terbaik. Hal ini dapat dilihat dari akurasi yang dihasilkan sebesar 100% untuk data *train* dan 93,57% untuk data *test.* Sedangkan dengan ukuran kernel 5x5 pada konvolusi lapisan pertama dan kedua, ukuran kernel 5x5 pada konvolusi lapisan pertama dan 2x2 pada konvolusi lapisan kedua, serta ukuran kernel 2x2 pada konvolusi lapisan pertama dan kedua menghasilkan akurasi dibawah 90%. Untuk ukuran kernel 5x5 pada konvolusi lapisan pertama dan 3x3 pada konvolusi lapisan kedua menghasilkan akurasi sebesar 100% untuk data *train* dan 90,00% untuk data *test.* Untuk ukuran kernel 3x3 pada konvolusi lapisan pertama dan kedua menghasilkan akurasi sebesar 100% untuk data *train* dan 91,14% untuk data *test.*

Selanjutnya adalah pengujian terhadap jumlah *epoch. Epoch* adalah iterasi yang dilakukan sistem untuk dapat mempelajari data *training*, sehingga jika sistem semakin baik dalam mempelajari data *training* maka semakin baik pula sistem dalam memprediksi data *testing.* Pada penelitian ini, jumlah *epoch* yang dibandingkan yaitu, 30, 50 dan 100. Berikut adalah akurasi dari hasil pengujian:

Tabel 5.4 Hasil pengujian jumlah *epoch*

Jumlah *Epoch*	Akurasi Data *Train*	Akurasi Data *Test*
30	100%	87,85%
50	**100%**	**93,57%**
100	100%	92,85%

Pengujian jumlah *epoch* dilakukan dengan nilai parameter ukuran kernel 3x3 untuk konvolusi pada lapisan pertama dan 2x2 untuk konvolusi lapisan kedua. Berdasarkan **gambar 5.11**, jumlah *epoch* 50 menghasilkan akurasi terbaik, dimana nilai akurasinya mencapai 100% untuk data *train* dan 93,57 untuk data *test.* Akurasi terendah dihasilkan dari jumlah *epoch* 30. Hal ini dikarenakan sistem belum cukup baik dalam mempelajari data *training* dengan iterasi sebanyak 30 maka sistem juga belum dapat memprediksi data *test* dengan sangat baik. Tetapi jika menggunakan jumlah *epoch* 100, akurasi data *test* berkurang sebesar 0,72%. Berikut adalah grafik yang menunjukkan akurasi hasil pengujian jumlah *epoch*:



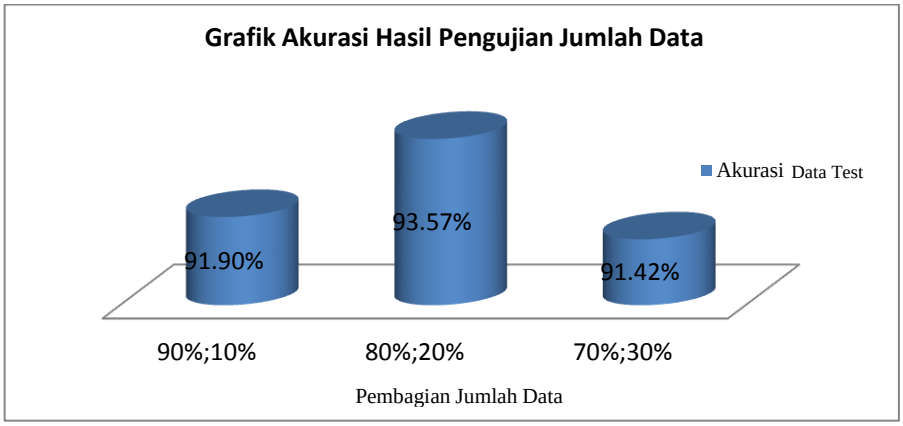
Gambar 5.11 Grafik akurasi pengujian *epoch*

Selanjutnya adalah pengujian terhadap jumlah data *train* dan *test* yang digunakan. Pada penelitian ini, pembagian jumlah data *train* dan data *test* yang dibandingkan yaitu, 70%;30%, 80%;20% dan 90%;20%. Berikut adalah akurasi dari hasil pengujian:

Tabel 5.5 Hasil pengujian jumlah data

Jumlah Data	Akurasi Data *Train*	Akurasi Data *Test*
70%;30%	100%	91,90%
80%;20%	**100%**	**93,57%**
90%;10%	100%	91,42%

Berdasarkan **tabel 5.5** di atas, dengan pembagian 80% data *train* dan 20% data *test* dari total data, menghasilkan akurasi terbaik, yaitu 100% untuk data *train* dan 93,57% untuk data *test.* Akurasi terendah dihasilkan dari pembagian 90% untuk data *train* dan 10% untuk data *test.* Hal ini dikarenakan data *test* hanya 10% dari total data atau 70 citra saja. Artinya data yang digunakan untuk data *test* kurang, sehingga sistem tidak dapat memprediksi data *test* secara baik. Berikut adalah grafik yang menggambarkan akurasi data *test* terhadap pengujian jumlah data:



Gambar 5.12 Grafik akurasi pengujian jumlah data

5.7 Hasil Klasifikasi CNN

Berdasarkan hasil pengujian parameter dalam penelitian ini untuk mendapatkan akurasi terbaik, diketahui bahwa ukuran kernel yang digunakan 3x3 untuk konvolusi lapisan pertama dan 2x2 untuk konvolusi lapisan kedua, jumlah *epoch* yang digunakan 50, pembagian data *train* dan data *test* sebesar 80% untuk data *train* (sebanyak 560 citra) dan 20% untuk data *test* (sebanyak 140 citra). Hasil pengujian parameter tersebut yang digunakan untuk klasifikasi CNN.

Tabel 5.6 adalah tabel hasil klasifikasi untuk data *train.* Pada masing-masing kategori pecahan uang kertas digunakan 80 data citra untuk *training.* Berdasarkan tabel tersebut, dapat diketahui bahwa sistem berhasil mengklasifikasikan semua citra uang kertas pecahan Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00, sedangkan untuk citra uang kertas pecahan Rp 10.000,00 berhasil diklasifikasikan sebanyak 76 citra dan 4 gagal.

Tabel 5.7 adalah tabel hasil klasifikasi untuk data *test.* Pada masing-masing kategori pecahan uang kertas digunakan 20 data citra untuk *testing.* Berdasarkan tabel tersebut, dapat diketahui bahwa sistem berhasil mengklasifikasikan semua citra uang kertas pecahan Rp 1.000,00, Rp 20.000,00, Rp 100.000,00, sedangkan untuk citra uang kertas pecahan Rp 2.000,00, Rp 10.000,00 dan Rp 50.000,00 berhasil diklasifikasikan sebanyak 18 citra dan 2 gagal serta untuk citra uang kertas pecahan Rp 5.000,00 berhasil diklasifikasikan sebanyak 17 citra dan 2 gagal. Berikut adalah **tabel 5.6** dan **tabel 5.7**:

Tabel 5.6 Hasil klasifikasi data *train*

Kategori	Tabel Klasifikasi							Berhasil		Gagal	
	Rp 1.000,00	Rp 2.000,00	Rp 5.000,00	Rp 10.000,00	Rp 20.000,00	Rp 50.000,00	Rp 100.000,00	Jumlah	Presentase	Jumlah	Presentase
Rp 1.000,00	80	0	0	0	0	0	0	80	100%	0	0%
Rp 2.000,00	0	80	0	0	0	0	0	80	100%	0	0%
Rp 5.000,00	0	0	80	0	0	0	0	80	100%	0	0%
Rp 10.000,00	0	0	4	76	0	0	0	76	95%	4	5%
Rp 20.000,00	0	0	0	0	80	0	0	80	100%	0	0%
Rp 50.000,00	0	0	0	0	0	80	0	80	100%	0	0%
Rp 100.000,00	0	0	0	0	0	0	80	80	100%	0	0%

Tabel 5.7 Hasil klasifikasi data *test*

Kategori	Tabel Klasifikasi							Berhasil		Gagal	
	Rp 1.000,00	Rp 2.000,00	Rp 5.000,00	Rp 10.000,00	Rp 20.000,00	Rp 50.000,00	Rp 100.000,00	Jumlah	Presentase	Jumlah	Presentase
Rp 1.000,00	20	0	0	0	0	0	0	20	100%	0	0%
Rp 2.000,00	0	18	1	1	0	0	0	18	90%	2	10%
Rp 5.000,00	1	0	17	0	1	0	1	17	85%	3	15%
Rp 10.000,00	0	0	0	18	0	2	0	18	90%	2	10%
Rp 20.000,00	0	0	0	0	20	0	0	20	100%	0	0%
Rp 50.000,00	0	2	0	0	0	18	0	18	90%	2	10%
Rp 100.000,00	0	0	0	0	0	0	20	20	100%	0	100%

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan penelitian dan hasil pengujian algoritma *Convolutional Neural Network* untuk klasifikasi uang kertas rupiah tahun emisi 2017, maka dapat diperoleh kesimpulan sebagai berikut:

1. Klasifikasi nominal uang kertas menggunakan algoritma *Convolutional Neural Network* dilakukan dengan jumlah data 700 citra, yang dibagi menjadi dua data *set*, yaitu 560 data *train* dan 140 data *test.* Proses *preprocessing* yang dilakukan pada citra adalah *resize* (mengubah ukuran citra menjadi 32x32) dan *grayscale* (mengubah warna citra menjadi hitam putih). Sistem dapat mengklasifikasikan nominal uang kertas rupiah tahun emisi 2017 menjadi 7 kategori, yaitu Rp 1.000,00, Rp 2.000,00, Rp 5.000,00, Rp 10.000,00, Rp 20.000,00, Rp 50.000,00 dan Rp 100.000,00.
2. Parameter yang digunakan adalah 2 lapisan konvolusi dengan ukuran kernel 3x3 pada lapisan pertama dan 2x2 pada lapisan konvolusi kedua, jumlah *filter* 20 pada lapisan konvolusi pertama dan kedua, probabilitas *dropout* sebesar 0,1, jumlah *epoch* 50, *batch size* 100 dan 80% dari total data untuk data *train* serta 20% dari total data untuk data *test*, sehingga memperoleh akurasi sebesar 100% untuk data *train* dan 93,57% untuk data *test.* Hal ini berarti bahwa, klasifikasi yang dihasilkan dari sistem termasuk ke dalam *excellent classification* karena menghasilkan akurasi yang tinggi.

6.2 Saran

Saran yang dapat diberikan untuk penelitian selanjutnya, khususnya yang menggunakan algoritma *Convolutional Neural Network* adalah:

1. Penelitian selanjutnya dapat menggunakan sinar UV untuk mengidentifikasi apakah uang tersebut asli ataupun palsu karena semakin maraknya kasus pemalsuan uang. Untuk mengimplementasikan sistem ini pada mesin penjualan otomatis (AVTM) perlu dilakukan peneletian lebih atau

sistem ini dapat diterapkan pada kasir sehingga tidak dibutuhkan banyak waktu pada saat proses pembayaran karena pengecekan uang dilakukan oleh mesin tidak secara manual.

2. Peneliti yang ingin menerapkan sistem ini pada mesin kasir dapat menambahkan kamera dan kotak pembatas *(bounding box)* agar sistem hanya mendeteksi objek yang diteliti (uang kertas) saja.
3. Penelitian selanjutnya dapat menggabungkan beberapa uang kertas dengan tahun emisi berbeda agar sistem dapat mengenali semua uang kertas rupiah yang masih beredar di masyarakat.
4. Penelitian selanjutnya dapat mencoba menggunakan uang kertas yang basah, terlipat ataupun sobek untuk mengetahui apakah sistem masih dapat mengenali uang kertas tersebut.
5. Penelitian selanjutnya dapat menggunakan perangkat GPU *(Graphical Processing Unit)* pada metode *Convolutional Neural Network* agar proses pengklasifikasian dilakukan secara cepat.

DAFTAR PUSTAKA

Anindita, Mediterasanti., Astri Novianty., M. Syaiful Sabril. 2012. Aplikasi Pendeteksi dan Pengenalan Nominal Uang Kertas Dengan Metode *Learning Vector Quantization* Sebagai Alat Bantu Bagi Penyandang Tunanetra. Jurnal IT Telkom, Volume 1, Nomor 2, September 2012, Halaman 137-144.

Danukusumo, Kefin Pudi. 2017. Implementasi *Deep Learning* Menggunakan *Convolutional Neural Network* Untuk Klasifikasi Citra Candi Berbasis GPU. Skripsi. Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Atma Jaya.

Gonzales, Rafael C., Richards E. Woods dan Steven L. Eddins. 2008. *Digital Image Processing Second Edition.* New Jersey:Prentice Hall

Han, J and Kamber, M. 2006. *Data Mining: Concepts and Techniques, Second Edition*, Morgan Kauffmann, San Fransisco.

Handayani, Hanny Hikmayanti. 2014. Penggunaan Algoritma *Backpropagation Levenberg Marquardt* dan Teknik Pengolahan Citra Digital Untuk Identifikasi Nominal Uang Kertas. Jurnal Ilmiah Solusi, Volume 1, Nomor 2, 2 April-Juni 2014, Halaman 16-33.

Hendra Heriawan. 2007. Pengenalan Mata Uang Rupiah Menggunakan Logika Fuzzy. Skripsi. Program Sarjana Fakultas Tekni Universitas Indonesia.

Hermawan, Arief. 2006. Jaringan Syaraf Tiruan: Teori dan Aplikasi. Yogyakarta:ANDI.

Kharisma, Gilang dan Achmad Subhan. 2011. Identifikasi Uang Kertas Menggunakan Metode *Backpropagasi* Pada Sistem Internet-Telpon Umum. Makalah Proyek Akhir. Politeknik Elektronika Negeri Surabaya ITS.

Krizhevsky, Alex., Ilya Sutskever., Geoffrey E. Hinton. 2012. *ImageNet Classification with Deep Convolutional Neural Networks.* NIPS'12 *Proceedings of the 25th International Conference on Neural Information Processing Systems* on December 03-06 2012, *Volume* 1, *Page* 1097-1105.

Kusumadewi, S. 2004. Membangun Jaringan Syaraf Tiruan. Yogyakarta: Graha Ilmu.

LeCun, Y., Bengio, Y dan Hinton. 2015. *Deep Learning. Nature* 521(7553):436-444.

LISA lab.2015. *Deep Learning* Tutorial. *Release* 0.1. University of Montreal.

Mark S. Nixon dan Alberto S. Aguado. *Feature Extraction in Computer Vision and Image Processingg.* Oxford : Newnes, 2002.

MathWorks. 2018. *Convolutional Neural Network.* U.SA

Ni'am, Bagus Asriel AA. 2013. Identifikasi Nilai Nominal dan Keaslian Uang Kertas Rupiah Menggunakan *Support Vector Machine.* Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Negeri Maulana Malik Ibrahim.

Nieslen, M. A. 2015. *Neural Networks and Deep Learning,* Determination Press.

Pramudiono. 2003. Pengantar Data Mining Menambang Permata Pengetahuan di Gunung Data. Kuliah Umum Ilmu Komputer.Com

Prasetyo, E. 2012. *Data Mining* Konsep dan Aplikasi Menggunakan Matlab, *1st ed.*, Andi Offset, Yogyakarta.

Pratama, Pius Juan. 2016. Identifikasi Nilai Nominal Uang Kertas Dengan Metode *Local Binary Pattern.* Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Sanata Dharma.

Putra, Wayan Suartika Eka., Arya Yudhi Wijaya., Rully Soelaiman. 2016. Klasifikasi Citra Menggunakan *Convolutional Neural Network* (CNN) Pada *Caltech* 101. Jurnal Teknik ITS, Volume 5, Nomor 1, Halaman 2337-3539.

Rajagede, R.A. 2016. *Deep Learning* Untuk Pengenalan Pelafalan Huruf Hijaiyah Berharakat. Skripsi. FMIPA Universitas Gadjah Mada.

Rahman, Fadhrul. 2011. Hukum Zakat Fitrah Menggunakan Uang Kertas. Skripsi. Konsentrasi Perbandingan Mazhab Fikih Program Studi Perbandingan Mazhab dan Hukum Fakultas Syariah dan Hukum Universitas Syarif Hidayatullah.

Ridzky Arya P. 2008. Perangkat Lunak Identifikasi Nilai Nominal dan Keaslian Mata Uang Rupiah Dengan *Image Recognition* Menggunakan *Hidden Markov* Model. Skripsi. Program Sarjana Fakultas Teknik Universitas Indonesia.

Rismiyati. 2016. Implementasi *Convolutional Neural Network* Untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital. Tesis. Program Studi S2 Ilmu Komputer Departemen Ilmu komputer dan Elektronika FMIPA Universitas Gadjah Mada.

Saputra, Indra Gunawan., Erwin Susanto., Ramadhan Nugraha. 2016. Implementasi Metode Jaringan Syaraf Tiruan (JST) Pada Alat Deteksi Nilai Nominal Uang. *E-Proceeding of Engineering: Volume* 3, *Nomor* 1, April 2016, *Page* 65.

Wicaksono, Dawud Gede. 2007. Perangkat Lunak Identifikasi Nominal dan Keaslian Uang kertas Rupiah Menggunakan Jaringan Syaraf Tiruan *Backpropagation.* Skripsi. Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia.

Zufar, Muhammad dan Budi Setiyono. 2016. *Convolutional Neural Networks* Untuk Pengenalan Wajah Secara *Real-Time*. Jurnal Sains dan Seni ITS, Volume 5, Nomor 2, Halaman 2337-3520.

LAMPIRAN

Lampiran 1. *Script Convolutional Neural Network* Menggunakan MXNet

```
#install package EBIMage
source ("https://bioconductor.org/biocLite.R")
biocLite("EBImage")
library(EBImage)
setwd("E://1 seribu")
# Set label
labelseribu <- 0
# Set up df
df <- data.frame()
# List images in path
seribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(seribu))
{

  uangseribu <- readImage(seribu[i])
  uangseribu_resized <- resize(uangseribu, w=w, h=h)
  ## Set to grayscale
  uangseribu_gray <- channel(uangseribu_resized, "gray")
  ## Get the image as a matrix
  uangseribu_matrix <- uangseribu_gray@.Data
  ## Coerce to a vector
  uangseribu_vector <- as.vector(t(uangseribu_matrix))
  seribuku=c(labelseribu,uangseribu_vector)
  df<-rbind(df,seribuku)
  write.csv(df, "seribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uangseribu_gray)


setwd("E://2 dua ribu")
# Set label
labelduaribu <- 1
# Set up df
df <- data.frame()
# List images in path
duaribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(duaribu))
{

  uangduaribu<- readImage(duaribu[i])
  uangduaribu_resized <- resize(uangduaribu, w=w, h=h)
```

```
  ## Set to grayscale
  uangduaribu_gray <- channel(uangduaribu_resized, "gray")
  ## Get the image as a matrix
  uangduaribu_matrix <- uangduaribu_gray@.Data
  ## Coerce to a vector
  uangduaribu_vector <- as.vector(t(uangduaribu_matrix))
  duaribuku=c(labelduaribu,uangduaribu_vector)
  df<-rbind(df,duaribuku)
  write.csv(df, "duaribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uangduaribu_gray )


setwd("E://3 lima ribu")
# Set label
labellimaribu <- 2
# Set up df
df <- data.frame()
# List images in path
limaribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(limaribu))
{

  uanglimaribu<- readImage(limaribu[i])
  uanglimaribu_resized <- resize(uanglimaribu, w=w, h=h)
  ## Set to grayscale
  uanglimaribu_gray <- channel(uanglimaribu_resized, "gray")
  ## Get the image as a matrix
  uanglimaribu_matrix <- uanglimaribu_gray@.Data
  ## Coerce to a vector
  uanglimaribu_vector <- as.vector(t(uanglimaribu_matrix))
  limaribuku=c(labellimaribu,uanglimaribu_vector)
  df<-rbind(df,limaribuku)
  write.csv(df, "limaribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display (uanglimaribu_gray)


setwd("E://4 sepuluh ribu")
# Set label
labelsepuluhribu <- 3
# Set up df
df <- data.frame()
# List images in path
sepuluhribu <- list.files()
#set widht
w<-32
#set height
```

```
h<-32
image_size<-32*32

for (i in 1:length(sepuluhribu))
{

  uangsepuluhribu<- readImage(sepuluhribu[i])
  uangsepuluhribu_resized <- resize(uangsepuluhribu, w=w, h=h)
  ## Set to grayscale
  uangsepuluhribu_gray <- channel(uangsepuluhribu_resized, "gray")
  ## Get the image as a matrix
  uangsepuluhribu_matrix <- uangsepuluhribu_gray@.Data
  ## Coerce to a vector
  uangsepuluhribu_vector <- as.vector(t(uangsepuluhribu_matrix))
  sepuluhribuku=c(labelsepuluhribu,uangsepuluhribu_vector)
  df<-rbind(df,sepuluhribuku)
  write.csv(df, "sepuluhribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uangsepuluhribu_gray)


setwd("E://5 dua puluh ribu")
# Set label
labelduapuluhribu <- 4
# Set up df
df <- data.frame()
# List images in path
duapuluhribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(duapuluhribu))
{

  uangduapuluhribu<- readImage(duapuluhribu[i])
  uangduapuluhribu_resized <- resize(uangduapuluhribu, w=w, h=h)
  ## Set to grayscale
  uangduapuluhribu_gray <- channel(uangduapuluhribu_resized,
  "gray")
  ## Get the image as a matrix
  uangduapuluhribu_matrix <- uangduapuluhribu_gray@.Data
  ## Coerce to a vector
  uangduapuluhribu_vector <- as.vector(t(uangduapuluhribu_matrix))
  duapuluhribuku=c(labelduapuluhribu,uangduapuluhribu_vector)
  df<-rbind(df,duapuluhribuku)
  write.csv(df, "duapuluhribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}

display(uangduapuluhribu_gray)
```

```
setwd("E://6 lima puluh ribu")
# Set label
labellimapuluhribu <- 5
# Set up df
df <- data.frame()
# List images in path
limapuluhribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(limapuluhribu))
{

  uanglimapuluhribu<- readImage(limapuluhribu[i])
  uanglimapuluhribu_resized <- resize(uanglimapuluhribu, w=w, h=h)
  ## Set to grayscale
  uanglimapuluhribu_gray <- channel(uanglimapuluhribu_resized,
  "gray")
  ## Get the image as a matrix
  uanglimapuluhribu_matrix <- uanglimapuluhribu_gray@.Data
  ## Coerce to a vector
  uanglimapuluhribu_vector <-
  as.vector(t(uanglimapuluhribu_matrix))
  limapuluhribuku=c(labellimapuluhribu,uanglimapuluhribu_vector)
  df<-rbind(df,limapuluhribuku)
  write.csv(df, "limapuluhribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uanglimapuluhribu_gray)


setwd("E://7 seratus ribu")
# Set label
labelseratusribu <- 6
# Set up df
df <- data.frame()
# List images in path
seratusribu <- list.files()
#set widht
w<-32
#set height
h<-32
image_size<-32*32

for (i in 1:length(seratusribu))
{

  uangseratusribu<- readImage(seratusribu[i])
  uangseratusribu_resized <- resize(uangseratusribu, w=w, h=h)
  ## Set to grayscale
  uangseratusribu_gray <- channel(uangseratusribu_resized, "gray")
  ## Get the image as a matrix
  uangseratusribu_matrix <- uangseratusribu_gray@.Data
```

```
  ## Coerce to a vector
  uangseratusribu_vector <- as.vector(t(uangseratusribu_matrix))
  seratusribuku=c(labelseratusribu,uangseratusribu_vector)
  df<-rbind(df,seratusribuku)
  write.csv(df, "seratusribuku.csv", row.names = FALSE)
  names(df) <- c('label', paste('pixel', c(1:image_size)))
}
display(uangseratusribu_gray)


# Load datasets
seribucsv<- read.csv("E://1 seribu/seribuku.csv")
duaribucsv<- read.csv("E://2 dua ribu/duaribuku.csv")
limaribucsv<- read.csv("E://3 lima ribu/limaribuku.csv")
sepuluhribucsv<- read.csv("E://4 sepuluh ribu/sepuluhribuku.csv")
duapuluhribucsv<- read.csv("E://5 dua puluh
ribu/duapuluhribuku.csv")
limapuluhribucsv<- read.csv("E://6 lima puluh
ribu/limapuluhribuku.csv")
seratusribucsv<- read.csv("E://7 seratus ribu/seratusribuku.csv")
dim(seribucsv)

#install package caret
install.packages("caret")
library(caret)
#Data partitions: randomly split 80% of data into training set
with equal weights for money, and the rest 20% will be used as the
test set.
## Bind rows in a single dataset
complete_set <-
rbind(seribucsv,duaribucsv,limaribucsv,sepuluhribucsv,duapuluhribu
csv,limapuluhribucsv,seratusribucsv)
## test/training partitions
training_index <- createDataPartition(complete_set$label, p = .8,
times = 1)
training_index <- unlist(training_index)
train_set <- complete_set[training_index,]
dim(train_set)
test_set <- complete_set[-training_index,]
dim(test_set)
#Reshape the data into a proper format required by the model:
## Fix train and test datasets
train_data <- data.matrix(train_set)
train_x <- t(train_data[, -1])
dim(train_x)
train_y <- train_data[,1]
dim(train_y)
train_array <- train_x
dim(train_array)
dim(train_array) <- c(32, 32, 1, ncol(train_x))

test_data <- data.matrix(test_set)
test_x <- t(test_set[,-1])
dim(test_x)
test_y <- test_set[,1]
test_array <- test_x
```

```
dim(test_array)
dim(test_array) <- c(32, 32, 1, ncol(test_x))

#install package mxnet
cran <- getOption("repos")
cran["dmlc"] <- "https://s3-us-west-2.amazonaws.com/apache-
mxnet/R/CRAN/"
options(repos = cran)
install.packages("mxnet")
library(mxnet)
require(mxnet)
#Training the model:
mx_data <- mx.symbol.Variable('data')
## 1st convolutional layer 3x3 kernel and 20 filters.
conv_1 <- mx.symbol.Convolution(data = mx_data, kernel = c(3, 3),
num_filter = 20)
tanh_1 <- mx.symbol.Activation(data = conv_1, act_type = "tanh")
pool_1 <- mx.symbol.Pooling(data = tanh_1, pool_type = "max",
kernel = c(2, 2), stride = c(2,2))
## 2nd convolutional layer 2x2 kernel and 20 filters.
conv_2 <- mx.symbol.Convolution(data = pool_1, kernel = c(2,2),
num_filter = 20)
tanh_2 <- mx.symbol.Activation(data = conv_2, act_type = "tanh")
pool_2 <- mx.symbol.Pooling(data = tanh_2, pool_type = "max",
kernel = c(2, 2),stride = c(2, 2))
## 1st fully connected layer
flat <- mx.symbol.Flatten(data = pool_2)
drop1 = mx.symbol.Dropout(data=flat, p=0.1)
fcl_1 <- mx.symbol.FullyConnected(data = drop1, num_hidden = 200)
tanh_3 <- mx.symbol.Activation(data = fcl_1, act_type = "tanh")
## 2nd fully connected layer
drop1 = mx.symbol.Dropout(tanh_3, p=0.1)
fcl_2 <- mx.symbol.FullyConnected(data = drop1, num_hidden = 7)
## Output
NN_model <- mx.symbol.SoftmaxOutput(data = fcl_2)


## Set seed for reproducibility
mx.set.seed(100)

## Device used. Sadly not the GPU :-(
device <- mx.cpu()

## Train samples
model <- mx.model.FeedForward.create(NN_model, X = train_array, y
                                     = train_y,
                                     ctx = device,
                                     num.round = 50,
                                     array.batch.size = 100,
                                     optimizer = "adam",
                                     eval.metric =
                                     mx.metric.accuracy,
                                     initializer = mx.init.Xavier
                                     (factor_type = "in", rnd_type
                                     = "gaussian", magnitude = 2),
                                     array.layout = "auto")
```

```
## test train set
predict_probs <- predict(model, train_array)
predict_probs
predicted_labels <- max.col(t(predict_probs))-1
table(train_data[, 1], predicted_labels)
### see accuracy of models
sum(diag(table(train_y, predicted_labels)))/560

## Test test set
predict_probs <- predict(model, test_array)
predict_probs
predicted_labels <- max.col(t(predict_probs))-1
table(test_data[, 1], predicted_labels)
### see accuracy of models
sum(diag(table(test_y, predicted_labels)))/140

#visualisasi Model
graph.viz(NN_model)
```