

FINETUNING MODEL LAYOUTLM UNTUK PEMBACAAN NOTA BERBAHASA INDONESIA

TUGAS AKHIR

Diajukan guna memenuhi sebagian persyaratan dalam rangka menyelesaikan
Pendidikan Sarjana Strata Satu (S1) Program Studi Teknologi Informasi



I Made Andre Dwi Winama Putra
NIM: 1905551003

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS UDAYANA
2022

FINETUNING MODEL LAYOUTLM UNTUK PEMBACAAN NOTA BERBAHASA INDONESIA

TUGAS AKHIR

Diajukan guna memenuhi sebagian persyaratan dalam rangka menyelesaikan
Pendidikan Sarjana Strata Satu (S1) Program Studi Teknologi Informasi



I Made Andre Dwi Winama Putra
NIM: 1905551003

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS UDAYANA
2022

PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Tugas Akhir ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di perguruan tinggi lain, dan sepanjang pengetahuan saya tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Denpasar, April 2023

I Made Andre Dwi Winama Putra

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadapan Ida Sang Hyang Widhi Wasa/Tuhan Yang Maha Esa, karena atas Asung Kerta Wara Nugraha-Nya, penulis dapat menyelesaikan tugas akhir dengan judul “*Finetuning Model LayoutLM untuk Pembacaan Nota berbahasa Indonesia*”. Selama pelaksanaan tugas akhir ini penulis mendapat banyak masukan dan bimbingan dari berbagai pihak. Untuk itu, penulis ingin mengucapkan rasa terima kasih kepada:

1. Bapak Ir. I Ketut Sudarsana, ST., Ph.D, selaku Dekan Fakultas Teknik universitas Udayana.
2. Bapak Dr. Eng. I Putu Agung Bayupati, ST.,MT., selaku Ketua Program Studi Teknologi Informasi Universitas Udayana.
3. Ibu Ni Kadek Ayu Wirdiani, ST., MT , selaku dosen pembimbing I dan Bapak Dr. A.A. Kompang Oka Sudana, S.Kom., MT, selaku dosen pembimbing II yang telah banyak memberikan masukan dan bimbingan selama penyusunan tugas akhir ini.
4. Bapak Gusti Made Arya Sasmita, ST., MT , selaku dosen pembimbing akademik, yang telah memberikan bimbingan selama menempuh pendidikan di Program Studi Teknologi Informasi Fakultas Teknik Universitas Udayana.
5. Teman-teman seperjuangan dan segenap civitas di Program Studi Teknologi Informasi Universitas Udayana yang telah memberikan sumbangan ide, pemikiran dan dukungan dalam penyusunan tugas akhir ini.

Penulis menyadari bahwa tugas akhir ini masih jauh dari sempurna. Akhir kata penulis memohon maaf jika ada kesalahan dalam penulisan tugas akhir ini.

Denpasar, April 2023

I Made Andre Dwi Winama Putra

ABSTRAK

Belanja adalah salah satu jenis transaksi yang menghasilkan sebuah catatan berbentuk nota pembayaran. Biasanya, nota diberikan dalam bentuk kertas kecil yang membuatnya mudah hilang. Penyimpanan informasi transaksi yang terdapat pada nota penting untuk dilakukan dalam bentuk digital. Penyimpanan dalam bentuk digital akan membuat informasi yang terkandung di dalamnya mudah diakses serta mengatasi permasalahan nota yang mudah hilang. Saat ini proses pemindahan informasi ke dalam bentuk digital masih dilakukan secara manual. Keberadaan sistem yang dapat mengekstrak informasi pada nota dapat mempercepat proses digitalisasi informasi. Penelitian ini mengusulkan sebuah sistem yang menerapkan *finetuning* pada Model LayoutLM serta dengan bantuan OCR dari *Google Vision* dapat digunakan untuk mengekstrak informasi transaksi yang terkandung dalam nota. *Finetuning* pada Model LayoutLM berhasil mendapatkan akurasi sebesar 97,98% dengan pengujian menggunakan data evaluasi serta akurasi sebesar 90% pada skenario uji dalam pembacaan nota berbahasa Indonesia. Namun, variasi pada bentuk nota yang beragam mempersulit proses ekstraksi informasi pada label yang saling berhubungan seperti nama produk, kuantitas, dan harga barang.

Kata Kunci: *Computer Vision, Optical Character Recognition, Android, Deep Learning, LayoutLM, Nota, Struk*

ABSTRACT

Shopping is a type of transaction that generates a record in the form of a payment receipt. Typically, the receipt is provided in the form of a small paper that can be easily lost. It is important to store the transaction information contained within the receipt into a digital form. By storing the information in digitally, it will make it easily accessible and it will overcome the problem of easily lost receipts. Currently, The process of transferring information into digital form is still being done manually. Having a system that can extract these information can help speeds up the digitalisation process tremendously. This research proposes a system that applies finetuning on LayoutLM Model and with the help of OCR from Google Vision can be used to extract transaction information contained in the receipt. Finetuning on the LayoutLM Model successfully achieved an accuracy of 97.98% on training data and 90% accuracy on realtime test scenario for reading receipts written in Indonesian language. However, variations in the diverse forms of receipts make the process of extracting information on interrelated labels such as product names, quantities, and prices difficult.

Kata Kunci: *Computer Vision, Optical Character Recognition, Android, Deep Learning, LayoutLM, Nota, Struk*

DAFTAR ISI

PERNYATAAN.....	iii
KATA PENGANTAR.....	iv
ABSTRAK	v
ABSTRACT	vi
DAFTAR ISI.....	vii
DAFTAR GAMBAR.....	x
DAFTAR KODE PROGRAM	xii
DAFTAR TABEL.....	xiii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	3
1.3 Tujuan Penelitian.....	3
1.4 Manfaat Penulisan	3
1.5 Batasan Masalah.....	4
1.6 Sistematika Penulisan.....	4
1.6.1 Bab I Pendahuluan	4
1.6.2 Bab II Tinjauan Pustaka.....	4
1.6.3 Bab III Metodologi Penelitian.....	5
1.6.4 Bab IV Pembahasan dan Analisis Hasil.....	5
1.6.5 Bab V Penutup	5
BAB II TINJAUAN PUSTAKA.....	6
2.1 State of the Art.....	6
2.2 Segmentasi.....	16
2.3 Layout Parser.....	17
2.4 OCR.....	17
2.5 Google Vision.....	18
2.6 LayoutLM.....	18
2.7 Metriks Evaluasi Model	19
BAB III METODOLOGI PENELITIAN	22
3.1 Tempat dan Waktu Penelitian	22
3.2 Data Penelitian.....	22

3.2.1	Data Primer	22
3.2.2	Data Sekunder	23
3.3	Instrumen Pembuatan Sistem	23
3.4	Alur Penelitian.....	24
3.4.1	Pembuatan Dataset.....	25
3.4.2	Finetuning Model LayoutLM.....	32
3.4.3	Evaluasi Model.....	33
3.4.4	Pengujian Sistem.....	34
3.5	Gambaran Umum Sistem	35
3.6	Alur Aplikasi	36
3.7	Rancangan Sistem	39
BAB IV HASIL DAN PEMBAHASAN.....		44
4.1	Pembuatan Dataset	44
4.1.1	Segmentasi Gambar	44
4.1.2	Pembacaan Karakter.....	52
4.1.3	Annotasi Dataset	53
4.2	Finetuning Model LayoutLM.....	56
4.2.1	Data Preparation.....	56
4.2.2	Data Pipelining.....	58
4.2.3	Model Finetuning.....	61
4.2.4	Model Evaluation.....	63
4.3	Deployment Sistem	66
4.3.1	Flask API Server	67
4.3.2	Aplikasi Android.....	68
4.4	Pengujian Sistem	70
4.4.1	Pengujian Inferensi Model	70
4.4.2	Pengujian Variasi Kecerahan Nota	77
4.4.3	Pengujian Variasi Panjang Nota.....	82
4.4.4	Hasil Pengujian Sistem	84
4.4.5	Kelebihan dan Kekurangan Sistem	85

BAB V PENUTUP.....	87
5.1 Kesimpulan.....	87
5.2 Saran.....	87
DAFTAR PUSTAKA	89
HALAMAN BELAKANG LAINNYA	93

DAFTAR GAMBAR

Gambar 2.1 Gambar Pelatihan LayoutLM (Xu et al., 2020)	19
Gambar 3.1 Flowchart Alur Penelitian	25
Gambar 3.2 Flowchart Pembuatan Dataset.....	26
Gambar 3.3 Proses Segmentasi ROI Nota	27
Gambar 3.4 Gambar Hasil Proses ROI	28
Gambar 3.5 Gambar Hasil Pembacaan Google Vision.....	29
Gambar 3.6 Gambar Hasil Proses Annotasi Dataset.....	31
Gambar 3.7 Flowchart Pelatihan LayoutLM	33
Gambar 3.8 Gambaran Umum Sistem	35
Gambar 3.9 Flowchart Pembacaan Nota pada Aplikasi Android	36
Gambar 3.10 Flowchart Proses Pembacaan Nota pada Web Server.....	37
Gambar 3.11 Flowchart Preprocessing Gambar pada Server	38
Gambar 3.12 Flowchart Melihat Histori Pembacaan Nota pada Aplikasi Android.....	39
Gambar 3.13 Rancangan Tampilan Depan Sistem	40
Gambar 3.14 Rancangan Halaman Scan.....	41
Gambar 3.15 Rancangan Hasil Preview Pembacaan	42
Gambar 3.16 Rancangan Riwayat Hasil Scan.....	43
Gambar 4.1 Hasil Proses Pembacaan Gambar Serta Resize	45
Gambar 4.2 Hasil Proses Bluring dan Dilasi Pada Gambar.....	46
Gambar 4.3 Hasil Deteksi Tepi Dengan Menggunakan Canny	47
Gambar 4.4 Deteksi Kontur Hasil tepi Canny	49
Gambar 4.5 Hasil Pemilihan Kontur	50
Gambar 4.6 Nota Hasil Segmentasi	51
Gambar 4.7 Hasil Deteksi Google Vision.....	52
Gambar 4.8 Gambar Penggunaan Fungsi Annotasi tahap 1	55
Gambar 4.9 Gambar Penggunaan Fungsi Annotasi tahap 2	55
Gambar 4.10 Hasil Segmentasi Bounding Box.....	58
Gambar 4.11 Hasil Pengolahan Dataset.....	59
Gambar 4.12 Hasil Pengubahan Data Menjadi Encoding.....	61
Gambar 4.13 Hasil Data Latih Classification Report.....	64
Gambar 4.14 Hasil Data Latih Confussion Matrix	65
Gambar 4.15 Hasil data uji Classification Report.....	65
Gambar 4.16 Hasil Data Uji Confussion Matrix.....	66
Gambar 4.17 Hasil Web Server Deployment.....	68
Gambar 4.18 Tampilan Home Aplikasi	68

Gambar 4.19 Tampilan Menu Sumber Gambar Nota	69
Gambar 4.20 Tampilan Proses Cropping Pada Aplikasi.....	69
Gambar 4.21 Tampilan Preview Hasil Pembacaan Nota	70
Gambar 4.22 Contoh Pengaruh Pencahayaan Pada Deteksi Sistem	81

DAFTAR KODE PROGRAM

Kode Program 4.1 Pembacaan File Gambar Nota	44
Kode Program 4.2 Preprocessing Nota Tahap 1	45
Kode Program 4.3 Proses Deteksi Tepi Canny	47
Kode Program 4.4 Deteksi Kontur Hasil tepi Canny	48
Kode Program 4.5 Pemilihan Kontur.....	49
Kode Program 4.6 Segmentasi Nota	51
Kode Program 4.7 Pembacaan Karakter Dengan Google Vision	52
Kode Program 4.8 Kode Program Otomasi Pembacaan OCR pada Setiap Nota.....	53
Kode Program 4.9 Fungsi Pembantu Annotasi Dataset	54
Kode Program 4.10 Persiapan Dataset.....	56
Kode Program 4.11 Fungsi Normalisasi Dataset	57
Kode Program 4.12 Segmentasi Kata Bounding Box	57
Kode Program 4.13 Inisiasi AutoProcessor LayoutLM.....	58
Kode Program 4.14 Pengolahan Dataset.....	59
Kode Program 4.15 Pengubahan Data Menjadi Encoding.....	60
Kode Program 4.16 Metrics Pelatihan Model	62
Kode Program 4.17 Memuat Model LayoutLM	63
Kode Program 4.18 Pelatihan Model	63
Kode Program 4.19 Evaluasi Model	64
Kode Program 4.20 Web Server Deployment.....	67

DAFTAR TABEL

Tabel 2.1 State of the Art	6
Tabel 2.2 Confusion Matrix	20
Tabel 3.1 Spesifikasi Perangkat Keras.....	23
Tabel 3.2 Spesifikasi Perangkat Lunak.....	24
Tabel 3.3 Pelabelan Dataset	30
Tabel 3.4 Daftar Sampel <i>Dataset</i> Primer	31
Tabel 4.1 Contoh Pengujian Inferensi Model	71
Tabel 4.2 Rangkuman Hasil Pengujian Inferensi Model	74
Tabel 4.3 Contoh Pengujian Kecerahan Nota	77
Tabel 4.4 Rangkuman Hasil Pengujian Kecerahan Nota	78
Tabel 4.5 Contoh Pengujian Gambar yang Tertutup Bayangan	79
Tabel 4.6 Rangkuman Hasil Pengujian Gambar tertutup bayangan	80
Tabel 4.7 Tabel Pengujian Variasi Panjang Nota	82

BAB I

PENDAHULUAN

Bab I pendahuluan pada laporan penelitian tugas akhir ini berisi tentang latar belakang, rumusan masalah, batasan masalah, manfaat, serta sistematika yang digunakan dalam pembuatan penelitian tugas akhir ini.

1.1 Latar Belakang

Transaksi adalah sebuah kesepakatan antara pembeli dan penjual untuk menukar barang dan jasa yang mereka miliki. Salah satu contoh dari transaksi adalah belanja. Belanja adalah salah satu jenis transaksi yang menghasilkan sebuah catatan berbentuk struk atau nota pembayaran. Belanja dilakukan dengan cara membayarkan sejumlah uang kepada penjual dan pihak penjual akan memberikan barang/jasa beserta struk atau nota belanja yang memuat isi dari transaksi yang dilakukan. Catatan dalam bentuk struk atau nota ini penting untuk disimpan agar pengeluaran dana dapat terlihat dengan jelas.

Penyimpanan informasi yang terdapat pada catatan belanja struk atau nota sebaiknya dilakukan dalam bentuk digital agar tidak mudah hilang. Pemindahan informasi transaksi ke dalam bentuk digital tentunya akan memakan banyak waktu jika setiap data transaksi tersebut harus di-*input* secara manual ke dalam komputer (Kumar, Kaware, & Singh, 2020). Keberadaan sistem yang dapat mengekstrak informasi pada struk atau nota dan menyimpannya dalam format digital secara otomatis akan meningkatkan efisiensi kerja. Pemindahan media penyimpanan ke dalam bentuk digital juga akan mengurangi risiko struk dan nota tersebut hilang. Penyimpanan informasi dalam bentuk digital membuat informasi tentang pengeluaran dalam rentang waktu tertentu lebih mudah terlihat. Salah satu metode yang dapat digunakan untuk mengekstrak informasi *text* adalah Metode *Optical Character Recognition*.

Optical Character Recognition (OCR) adalah proses konversi gambar huruf menjadi karakter ASCII yang dikenali oleh komputer. Teknologi OCR dapat mengubah gambar yang berasal dari dokumen yang di-*scan*, tulisan digital, maupun tulisan tangan (Mohammad et al., 2014). Proses ekstraksi informasi menggunakan *Optical Character Recognition* (OCR) saat ini sudah memiliki tingkat akurasi yang tinggi yaitu hingga 95%, baik dalam pengenalan karakter digital (*digital character*) maupun dalam pengenalan tulisan tangan (*handwritten character*). Penerapan teknologi OCR dapat digunakan untuk membuat proses ekstraksi informasi yang terdapat pada struk dan nota dapat dilakukan secara otomatis (Kumar, Kaware, Singh, et al., 2020).

Teknologi OCR dapat digunakan untuk mendeteksi kalimat pada struk dan nota. Penerapan OCR dalam mendeteksi struk dan nota akan mengurangi waktu yang dibutuhkan untuk pemindahan informasi belanja menjadi bentuk digital. Manfaat lainnya dari keberadaan aplikasi OCR struk dan nota ini adalah total pengeluaran dalam jangka waktu tertentu dapat terlihat dengan mudah tanpa perlu menghitungnya secara manual.

Sistem pembacaan bukti transaksi berbentuk nota sudah pernah dibuat dengan menggunakan Metode *Template Matching* serta *rough positioning* oleh Lin C. Penggunaan Metode *Template Matching* dan *rough positioning* memiliki kekurangan yaitu setiap nota yang dideteksi harus sesuai dengan standar yang telah ditetapkan. LayoutLM adalah sebuah Model *Deep Learning* baru yang dapat berlatih pada *text* serta *bounding box* sebagai letak dari *text* tersebut. Penggunaan Model LayoutLM untuk mendeteksi nota diharapkan membuat pembacaan informasi dapat dilakukan pada bentuk nota yang beragam (Lin et al., 2022).

Sistem pembacaan bukti transaksi berbentuk nota bekerja dengan cara mengakuisisi citra bukti transaksi yang diambil menggunakan kamera dari *smartphone* Android. Citra tersebut selanjutnya dikirim menuju *web server* dengan menggunakan *API Call*. *Web server* akan menerima dan memproses citra bukti tersebut hingga menjadi *text* dan mengirimkannya kembali kepada *smartphone*

Android. Informasi tentang bukti pembayaran yang terdeteksi akan ditampilkan dalam layar *smartphone* dan pengguna dapat memilih untuk menyimpan bukti transaksi tersebut.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka masalah-masalah yang dapat dirumuskan adalah sebagai berikut.

1. Bagaimana cara merancang dan membangun aplikasi yang dapat membaca bukti transaksi berbentuk kertas menjadi bentuk digital secara otomatis.
2. Bagaimanakah akurasi dari sistem pembacaan nota belanja yang dibuat.

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang didapatkan, tujuan yang ingin dicapai dari penelitian tugas akhir ini adalah sebagai berikut.

1. Menghasilkan sebuah aplikasi yang dapat membaca dan menyimpan bukti transaksi dalam bentuk digital secara otomatis.
2. Menghasilkan sistem yang dapat mengekstrak informasi dalam bukti transaksi dengan akurasi yang baik.

1.4 Manfaat Penulisan

Manfaat penelitian ini dibuat berdasarkan masalah yang ingin diselesaikan, serta tujuan ingin dicapai oleh penulis. Penelitian ini diharapkan dapat memberikan manfaat kepada Pengguna dari aplikasi. Pengguna aplikasi dapat merasakan manfaat dari kemudahan pencatatan pembayaran hanya dengan menggunakan gambar serta dapat memahami dan melakukan manajemen keuangan dengan lebih baik.

1.5 Batasan Masalah

Batasan masalah yang digunakan dalam penelitian ini dibuat untuk memastikan agar ruang lingkup penelitian tidak terlampau jauh dan melebar. Batasan masalah dari penyusunan laporan tugas akhir ini adalah sebagai berikut.

1. *Dataset* yang digunakan berasal dari berbagai nota belanja dari minimarket dan restoran yang berupa nota *print out*.
2. Nota yang digunakan berbentuk persegi panjang dan tidak terlipat.
3. Sistem hanya akan membaca satu buah nota pada setiap gambar yang dikirimkan ke *web server*.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan pada laporan tugas akhir ini terdiri dari pendahuluan, tinjauan pustaka, pembahasan metodologi penelitian, hasil pembuatan sistem, serta simpulan dan saran yang dirangkum secara urut dan sistematis.

1.6.1 Bab I Pendahuluan

Bab I dimulai dari penjelasan mengenai latar belakang diambilnya topik penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, hingga sistematika penulisan yang digunakan untuk menyusun penelitian ini.

1.6.2 Bab II Tinjauan Pustaka

Bab II berisikan tentang *State of the Art* (SOTA) dari permasalahan yang diangkat, serta dasar-dasar teori yang bersumber dari artikel-artikel akademik dan platform pembelajaran akademik. Dasar-dasar teori ini kemudian digunakan sebagai dasar dalam pembahasan permasalahan dan solusi penelitian yang dilakukan.

1.6.3 Bab III Metodologi Penelitian

Bab III metodologi penelitian memuat informasi mengenai tempat dan waktu pelaksanaan penelitian, sumber data yang digunakan dalam penelitian, serta alur penelitian. Bab ini juga membahas mengenai instrumen pembuatan sistem serta rancangan dari sistem yang akan dibuat.

1.6.4 Bab IV Pembahasan dan Analisis Hasil

Bab IV berisikan informasi tentang pembahasan hasil dari penelitian mulai dari hasil pengumpulan data yang dilakukan, proses pemodelan, hingga evaluasi Model. Bab ini juga menampilkan sistem yang dibuat serta pembahasan mengenai hasil penelitian secara keseluruhan.

1.6.5 Bab V Penutup

Bab V berisikan mengenai simpulan dan saran yang dibuat selama dilaksanakannya penelitian ini. Simpulan mengacu pada hasil yang didapatkan selama pengerjaan pembuatan sistem. Saran berisikan tentang masukan yang dapat diberikan kepada pembaca untuk mendapatkan hasil yang lebih baik, maupun untuk mempermudah replikasi dari penelitian.

BAB II

TINJAUAN PUSTAKA

Bab II ini berisikan tentang dasar-dasar teori yang bersumber dari artikel-artikel akademik, serta platform pembelajaran akademik yang digunakan sebagai dasar dalam pembuatan *finetuning* pada Model LayoutLM ini.

2.1 State of the Art

State of the art adalah kumpulan artikel, jurnal dan literatur yang membahas topik serupa dengan penelitian ini. Pembuatan *state of the art* dilakukan dengan tujuan untuk membandingkan teknologi yang saat ini digunakan untuk menyelesaikan masalah yang serupa. Masalah yang dibahas pada penelitian ini adalah *Natural Language Processing* (NLP) dengan menggunakan *Optical Character Recognition* (OCR).

Tabel 2.1 State of the Art

No	Jurnal	Keterangan
1	<p>Judul <i>Universal Language Model Fine-tuning for Text Classification</i></p> <p>Peneliti Howard, Jeremy Ruder, Sebastian</p> <p>Tahun 2018</p> <p>Publikasi <i>Proceedings of the 56th</i></p>	<p>Howard menyampaikan dalam artikelnya bahwa sampai saat ini penyelesaian masalah NLP masih bersifat spesifik pada satu permasalahan. Penyelesaian yang bersifat spesifik pada satu permasalahan membuat pembuatan arsitektur baru serta pelatihan ulang harus dilakukan agar Model dapat menyelesaikan permasalahan baru. Penyelesaian yang bersifat spesifik ini membuat Howard membuat Model bernama <i>Universal Language Model Fine-tuning</i> (ULMFiT). ULMFiT adalah sebuah Model yang dirancang</p>

	<p><i>Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)</i></p>	<p>dan dikembangkan dengan tujuan agar proses <i>Transfer Learning</i> dapat dilakukan dengan lebih efektif dan dapat digunakan untuk menyelesaikan berbagai permasalahan NLP. Hasilnya Howard berhasil membuat Model ULMFiT dengan hasil akurasi yang setara dengan berbagai Model SOTA pada kategori permasalahan NLP yang berbeda (Howard & Ruder, 2018).</p>
2	<p>Judul <i>BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding</i></p> <p>Peneliti Devlin, Jacob Chang, Ming-Wei Lee, Kenton Toutanova, Kristina</p> <p>Tahun 2019</p> <p>Publikasi <i>Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language</i></p>	<p>Devlin melanjutkan perkembangan penggunaan <i>transfer learning</i> pada permasalahan NLP dan membuat sebuah Model bernama <i>Bidirectional Encoder Representations from Transformers</i> (BERT). BERT adalah sebuah <i>pretrained</i> Model yang menggunakan representasi <i>Deep Bidirectional</i> dari teks dengan cara menggabungkan konteks kiri dan kanan dari teks tersebut pada setiap <i>layer</i>-nya. Hasil pembuatan Model BERT ini berhasil mendapatkan akurasi di atas 82.1% yang berhasil melampaui Model SOTA dalam berbagai <i>dataset</i>. Model BERT yang dihasilkan juga dapat dengan mudah di-<i>tuning</i> untuk menyelesaikan berbagai jenis permasalahan NLP seperti <i>language inference</i>, dan <i>question answering</i> tanpa mengubah banyak dari arsitektur Model (Devlin et al., 2018).</p>

	<i>Technologies, Volume 1 (Long and Short Papers)</i>	
3	<p>Judul <i>BioBERT: a pre-trained biomedical language representation model for biomedical text mining</i></p> <p>Peneliti Jinhyuk Lee Wonjin Yoon Sungdong Kim Donghyeon Kim Sunkyu Kim, Chan Ho So, Jaewoo Kang</p> <p>Tahun 2019</p> <p>Publikasi <i>Bioinformatics, Volume 36, Issue 4, February 2020</i></p>	<p>Penelitian tentang <i>Finetuning</i> dari Model BERT, pernah digunakan untuk memahami literatur biomedis yang terus berkembang dengan sangat cepat. Setiap harinya terdapat sekitar 3000 artikel baru tentang literatur biomedis yang membuat ekstraksi informasi biomedis memerlukan <i>tools</i> yang akurat. Hasil penerapan <i>Finetuning</i> pada Model BERT berhasil membuat model bernama BioBERT dan mendapatkan akurasi mencapai 72 % dalam berbagai <i>dataset</i> serta berhasil melampaui Model <i>state of the art</i> dalam beberapa kategori ekstraksi informasi biomedis (Lee et al., 2020).</p>
4	<p>Judul <i>Automatic Receipt Recognition System Based on Artificial Intelligence Technology</i></p> <p>Peneliti Lin, Cheng Jian Liu, Yu Cheng</p>	<p>Metode pembacaan OCR pada nota belanja pernah dilakukan dengan mengaplikasikan <i>Template Matching</i> pada nota yang ingin dibaca. Hasil dari penggunaan Metode <i>Template Matching</i> tersebut selanjutnya akan dibaca dengan menggunakan Model <i>Deep Learning YOLOv4-s</i>. Sistem yang dikembangkan ini berhasil mendapatkan akurasi sebesar 80.93% dengan</p>

	<p>Lee, Chin Ling</p> <p>Tahun</p> <p>2021</p> <p>Publikasi</p> <p><i>Applied Sciences, Volume 12, Issue 2 (January-2022)</i></p>	<p>percobaan menggunakan CNN dan sebesar 99.39% dengan menggunakan <i>Yolov4-s</i> yang dikembangkan dalam pembacaan karakter (Lin et al., 2022).</p>
5	<p>Judul</p> <p>Implementasi Tesseract OCR untuk Pembuatan Aplikasi Pengenalan Nota pada Android</p> <p>Peneliti</p> <p>Andreas, Yoel Gunadi, Kartika Purbowo, Anita Nathania</p> <p>Tahun</p> <p>2020</p> <p>Publikasi</p> <p>Jurnal Infra (2020) vol 8</p>	<p>Pembacaan nota pernah dilakukan dengan memanfaatkan <i>Library BlinkReceipt</i>. <i>BlinkReceipt</i> merupakan sebuah <i>Application Programming Interface</i> (API) yang dapat digunakan dalam platform <i>IOS, Android</i>, serta melalui <i>javascript</i>. <i>Library BlinkReceipt</i> adalah sebuah <i>library</i> yang dapat digunakan secara khusus untuk membaca nota belanja. Hasil yang didapatkan dalam penggunaan <i>Library BlinkReceipt</i> memiliki beberapa kekurangan seperti aplikasi tidak dapat membaca potongan harga, serta pembacaan harus dilakukan berulang-ulang jika jumlah daftar belanja dalam struk cukup banyak (Andreas et al., 2020).</p>
6	<p>Judul</p> <p><i>Deep Learning For Automatic Sale Receipt Understanding</i></p> <p>Peneliti</p> <p>Raoui-Outach, Rizlene</p>	<p>Pembacaan merek toko dengan menerapkan lokalisasi pada gambar nota belanja pernah dilakukan pada tahun 2018. Lokalisasi gambar nota belanja ini dilakukan dengan cara mendeteksi tepi yang digunakan untuk mengklasifikasi merek toko. Pembacaan OCR</p>

	<p>Million-Rousseau Benoit, Alexandre Lambert, Patrick</p> <p>Tahun 2017</p> <p>Publikasi <i>Proceedings of the 7th International Conference on Image Processing Theory, Tools and Applications, IPTA 2017</i></p>	<p>dari merek toko dilakukan dengan <i>Deep Convolutional Neural Networks</i> (CNN). Hasil dari percobaan ini berhasil melokalisasi merek toko sebesar 86% (Raoui-Outach et al., 2018).</p>
7	<p>Judul Pemanfaatan <i>Optical Character Recognition</i> dan <i>Text Feature Extraction</i> untuk Membangun Basis Data Pengaduan Tenaga Kerja</p> <p>Peneliti Puspitarani, Yan Syukriyah, Yenie</p> <p>Tahun 2020</p> <p>Publikasi JURNAL RESTI Vol. 4 No. 4 (2020)</p>	<p>Pembacaan pengaduan ketenagakerjaan dengan menggunakan <i>Tesseract OCR</i> dan <i>NTLK toolkit</i> pernah dilakukan untuk mengekstrak informasi pada surat. Penelitian ini menggunakan <i>NTLK toolkit</i> untuk memahami isi surat, sedangkan proses ekstraksi tulisan surat dilakukan dengan menggunakan <i>Tesseract OCR</i>. Penelitian ini berhasil mengekstrak informasi sebesar 66.7% pada surat tulisan tangan dan sebesar 91,67% pada surat yang diketik (Puspitarani & Syukriyah, 2020).</p>

8	<p>Judul <i>Segmentation for document layout analysis: not dead yet</i></p> <p>Peneliti Markewich, Logan Zhang, Hao Xing, Yubin Lambert-Shirzad, Navid Jiang, Zhexin</p> <p>Lee, Roy Ka-Wei Li, Zhi Ko, Seok-Bum</p> <p>Tahun 2020</p> <p>Publikasi <i>International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25</i></p>	<p>Penggunaan Metode <i>Weighted Bounding Box Regression Loss</i> untuk melakukan segmentasi pada dokumen pernah dilakukan untuk meningkatkan akurasi. Penggunaan metode ini membuat hasil segmentasi dokumen yang memiliki banyak objek-objek kecil dapat dilakukan dengan lebih akurat. Hasil yang didapatkan dalam pengujian segmentasi pada <i>Dense Article Dataset (DAD)</i> dan <i>dataset PubLayNet</i> memiliki <i>f1 score</i> sebesar 96.26% dan 97.11% dengan menggunakan Model <i>DeeplabV3+</i> (Markewich et al., 2022).</p>
9	<p>Judul <i>TableSegNet: a fully convolutional network for table detection and segmentation in document images</i></p> <p>Peneliti Nguyen, Duc-Dung</p> <p>Tahun</p>	<p>Model <i>TableSegNet</i> merupakan Model segmentasi yang dapat mendeteksi tabel dalam sebuah gambar dokumen. <i>TableSegNet</i> menggunakan arsitektur <i>fully convolutional network</i> untuk mendeteksi dan membedakan tabel secara bersamaan. Hasilnya penelitian Model segmentasi <i>TableSegNet</i> ini dapat menghasilkan akurasi sebesar 90% dalam dataset ICDAR2019 (Nguyen, 2022).</p>

	2022 Publikasi <i>International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25</i>	
10	Judul <i>MRZ code extraction from visa and passport documents using convolutional neural networks</i> Peneliti Liu, Yichuan James, Hailey Gupta, Otkrist Raviv, Dan Tahun 2022 Publikasi <i>International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25</i>	<p>Ekstraksi kode MRZ pada dokumen <i>visa</i> dan <i>passport</i> pernah dilakukan dengan menggunakan <i>Convolutional Neural Networks (CNN)</i>. Model CNN dibuat untuk mendeteksi <i>MRZ code</i> dari gambar <i>passport</i> digital. Hasilnya Model ini dapat mendeteksi 100% dari kode MRZ dan 99.25% <i>macro-f1</i> dari pengenalan karakter pada <i>dataset passport</i> dan Visa (Liu et al., 2022).</p>
11	Judul <i>Learning from similarity and information extraction from structured documents</i>	<p>Penelitian tentang ekstraksi informasi pada dokumen yang terstruktur pernah dilakukan dengan menggunakan arsitektur <i>Siamese Networks</i>. Penggunaan arsitektur <i>Siamese</i></p>

	<p>Peneliti Holeček, Martin</p> <p>Tahun 2021</p> <p>Publikasi <i>International Journal on Document Analysis and Recognition (IJDAR) (2021) vol 24</i></p>	<p><i>Networks</i> dikombinasikan dengan Metode <i>Similarity</i>, <i>One-Shot Learning</i>, dan <i>Context/Memory Awareness</i> untuk melakukan proses ekstraksi informasi pada dokumen. Hasil dari arsitektur pada penelitian ini berhasil meningkatkan skor f1 sebesar 8.25% bila dibandingkan dengan arsitektur <i>query answer</i> (Holeček, 2021).</p>
12	<p>Judul <i>Implementation of Zoning and K-Nearest Neighbor in Character Recognition of Wrésastra Script</i></p> <p>Peneliti Darma, I Wayan Agus Surya</p> <p>Tahun 2019</p> <p>Publikasi Lontar Komputer : Jurnal Ilmiah Teknologi Informasi (2019)</p>	<p>Penelitian tentang pembacaan karakter huruf Bali pada <i>wrésastra script</i> dan karakter huruf Bali pernah dilakukan dengan menggunakan <i>Zoning Feature Extraction</i> serta <i>K-Nearest Neighbor</i> (KNN). Penggunaan <i>Zoning Feature Extraction</i> dilakukan untuk mengekstrak fitur dari setiap karakter dan menjadikannya <i>embedding</i> untuk diklasifikasikan dengan Model KNN. Hasil yang pada penelitian ini adalah Model terbaik didapatkan saat parameter <i>neighbor</i> yang digunakan adalah tiga dengan akurasi mencapai 97.5% (Darma, 2019).</p>
13	<p>Judul <i>Handwriting Identification Using Deep Convolutional Neural Network Method</i></p> <p>Peneliti Sudana, Oka</p>	<p>Penelitian tentang klasifikasi karakter penulisan tangan pernah dilakukan dengan menggunakan <i>Convolutional Neural Network</i> (CNN). Arsitektur CNN digunakan untuk mengklasifikasi tulisan tangan adalah VGG19 dengan layer <i>output</i> yang telah dimodifikasi.</p>

	<p>Gunaya, I. Wayan Putra, I. Ketut Gede Darma Tahun 2020 Publikasi <i>Telecommunication Computing Electronics and Control (2020) vol 18</i></p>	<p>Hasil dari penelitian ini mendapatkan 90% akurasi dengan menggunakan gambar <i>grayscale</i> dan berhasil mengklasifikasi karakter penulisan orang dengan benar (Sudana et al., 2020).</p>
14	<p>Judul <i>Detection of fake news using deep learning CNN–RNN based methods</i> Peneliti Sastrawan, I. Kadek Bayupati, I. P.A. Arsa, Dewa Made Sri Tahun 2020 Publikasi ICT Express (2022) vol 8</p>	<p>Penelitian tentang deteksi berita bohong pernah dilakukan dengan menggunakan Model CNN, <i>Bidirectional LSTM</i>, dan <i>ResNet</i>. Proses pengubahan kata menjadi <i>token</i> pada penelitian ini dilakukan dengan menggunakan <i>embedding Word2Vec</i>, <i>GloVe</i>, dan <i>FastText</i>. Hasil dari penelitian ini mendapatkan kombinasi <i>embedding GloVe</i> dengan arsitektur <i>Bidirectional LSTM</i> memberikan hasil yang terbaik dengan akurasi melampaui 94.6%. (Sastrawan et al., 2022).</p>
15	<p>Judul <i>Balinese Script Recognition Using Tesseract Mobile Framework</i> Peneliti Indrawan, Gede Asroni, Ahmad Joni Erawati Dewi, Luh</p>	<p>Penelitian tentang pengenalan karakter bahasa Bali pernah dilakukan dengan menggunakan <i>Tesseract OCR 5</i>. Pelatihan Model <i>Tesseract OCR 5</i> dilakukan menggunakan <i>tools</i> pelatihan <i>Tesseract OCR</i>. Penggunaan <i>tools</i> pelatihan <i>Tesseract OCR</i> dilakukan untuk mengekstrak dan mengenali karakter bahasa Bali yang terdeteksi pada sistem. Penelitian ini berhasil</p>

	<p>Gunadi, I Gede Aris Paramarta, I Ketut</p> <p>Tahun 2020</p> <p>Publikasi Lontar Komputer : Jurnal Ilmiah Teknologi Informasi (2022) vol 13</p>	<p>mendapatkan skor <i>coincidence</i> sebesar 66.67% dengan memperhitungkan hierarki dari masing-masing karakter, kata, kalimat, serta paragraf dari penulisan bahasa Bali (Indrawan et al., 2022).</p>
16	<p>Judul <i>Research on a handwritten character recognition algorithm based on an extended nonlinear kernel residual network</i></p> <p>Peneliti Zheheng Rao Chunyan Zeng Minghu Wu Zhifeng Wang Nan Zhao Min Liu Xiangkui Wan</p> <p>Tahun 2018</p> <p>Publikasi <i>KSII Transactions on Internet and Information Systems</i> (2018) vol 12</p>	<p>Penelitian tentang pengenalan tulisan tangan ini dilakukan untuk meneliti sebuah metode yang dapat mengurangi proses komputasi model OCR. Penelitian ini memberikan pernyataan bahwa model <i>deep learning</i> yang menggunakan <i>extremely deep network</i> dalam melakukan OCR memerlukan kemampuan komputasi yang tinggi. Solusi yang diberikan oleh penelitian ini untuk mempercepat proses pelatihan model adalah dengan menggunakan <i>Extended Nonlinear Kernel Residual Network</i>. Hasil yang didapatkan adalah model dapat mencapai pengenalan karakter hingga 97.72% dengan waktu latih yang lebih singkat dari model berbasis <i>Residual Network</i> maupun model berbasis CNN (Rao et al., 2018).</p>

17	<p>Judul <i>Segmentation based, omnifont printed Arabic character recognition without font identification</i></p> <p>Peneliti Aziz Qaroush Abdalkarim Awad Mohammad Modallal Malik Ziq</p> <p>Tahun 2022</p> <p>Publikasi <i>Journal of King Saud University - Computer and Information Sciences</i> (2022) vol 34</p>	<p>Penelitian ini memberikan sebuah metode untuk melakukan pengenalan karakter tanpa menggunakan proses pengenalan <i>font</i> dari karakter. Pendekatan yang dilakukan adalah dengan menggunakan <i>indirect character segmentation</i>. Hasil akurasi rata-rata yang didapatkan dari penelitian ini adalah 95% tanpa pengenalan tipe <i>font</i> ataupun teknik <i>post processing</i> lainnya (Qaroush et al., 2022).</p>
----	--	--

Tabel 2.1 merupakan tabel *state of the art* yang digunakan sebagai referensi dalam pelaksanaan penelitian ini. Tabel *state of the art* ini memuat kumpulan penelitian yang berasal dari jurnal dan konferensi yang menjadi dasar dalam menentukan alur penelitian serta pemilihan metode dalam penelitian ini.

2.2 Segmentasi

Segmentasi adalah sebuah teknik untuk memisahkan citra menjadi beberapa *region* yang berbeda (Ngurah et al., 2019). Proses segmentasi sering digunakan dalam memproses sebuah citra hasil foto, agar dapat dibedakan antara wilayah *foreground* serta wilayah *background*. Selain itu, Segmentasi juga dapat digunakan untuk

menandai area dengan tulisan/kata yang terdapat pada gambar. Segmentasi dapat dilakukan secara otomatis maupun manual. Segmentasi secara otomatis dilakukan dengan melatih sebuah *neural network* untuk dapat memahami objek segmentasi (Smith et al., 2023).

2.3 Layout Parser

Layout Parser adalah sebuah *tools* yang dapat digunakan untuk melakukan proses segmentasi pada sebuah gambar dokumen. *Layout Parser* berfungsi sebagai alat pembantu proses ekstraksi *layout* dari gambar nota sehingga area penting dalam gambar nota dapat dibaca dengan menggunakan OCR. *Layout Parser* tersedia untuk digunakan dalam *Python* melalui *library Layout Parser* (Shen et al., 2021).

2.4 OCR

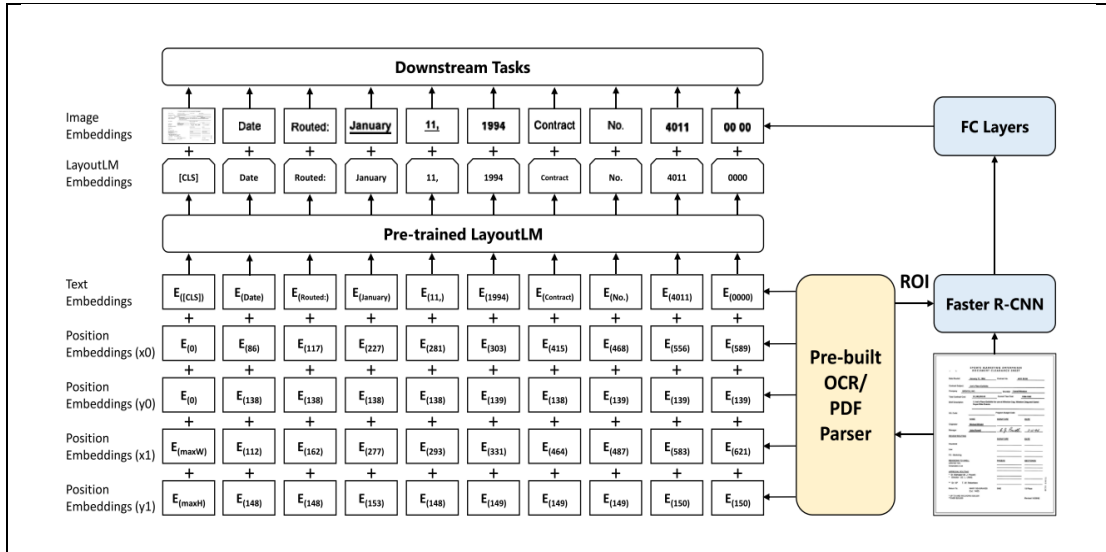
Optical Character Recognition (OCR) adalah proses konversi *text* dalam citra gambar menjadi format *text* yang dapat dibaca oleh mesin. Gambar yang digunakan untuk proses konversi *text* dapat berasal dari *text* yang dicetak maupun *text* hasil tulisan tangan (*handwritten character*) (Memon et al., 2019). Teknologi OCR merupakan bagian dari *artificial intelligence* yang banyak digunakan dalam bidang otomasi seperti pemindaian dokumen dan kuesioner, pembacaan pelat nomor kendaraan, verifikasi dokumen, dll (Qaroush et al., 2022). Beberapa arsitektur Model yang dapat digunakan sebagai dasar untuk melakukan OCR adalah Model *Long Short Term Memory* (LSTM), *Convolutional Neural Network* (CNN), dll. Model OCR sering dipadukan dengan Model *Natural Language Processing* (NLP) untuk meningkatkan akurasi dari pembacaan *text* (Hajiali et al., 2022). Arsitektur NLP yang sering digunakan dalam hal ini adalah *Bidirectional Encoder Representations from Transformers* (BERT).

2.5 Google Vision

Google Vision API merupakan sebuah Model *machine learning* yang telah dilatih untuk melakukan deteksi OCR melalui REST serta RPC API (Google Cloud, 2022). *Google Vision* API dapat melakukan *annotasi* pada gambar dan memberikan label pada masing-masing kategori yang terdeteksi pada gambar. Proses *annotasi* ini disebut sebagai proses *automatic image annotation* (Baker & Collins, 2023). *Automatic image annotation* dari *Google Vision* API dapat mengekstrak konten dari sebuah gambar untuk mendapatkan informasi visual seperti: memberikan label pada gambar, mendeteksi *landmark* wajah, OCR, dll (Saputra et al., 2019). Penggunaan *Google Vision* pada penelitian ini digunakan untuk mendeteksi setiap *text* yang ada pada sebuah nota serta letak *bounding box* dari *text* tersebut.

2.6 LayoutLM

LayoutLM merupakan sebuah Model *document understanding* yang dibuat untuk dapat memahami struktur dari sebuah dokumen. Model ini adalah dibuat dengan memperhatikan perkembangan permasalahan *Natural Language Processing* (NLP), di mana pada setiap Model NLP selalu berfokus pada *text-level manipulation*. Model LayoutLM ini dibuat dengan menggunakan interaksi antar informasi pada *teks* dalam sebuah dokumen beserta *layout* dari dokumen tersebut. Pengembangan Model ini dilakukan dengan menggunakan data dari dokumen-dokumen yang telah di-*scan* yang berasal dari berbagai macam kategori seperti surat, *memo*, *email*, *invoice*, *news*, *articles*, *questionnaire*, *resume*, dll (Xu et al., 2020).



Gambar 2.1 Gambar Pelatihan LayoutLM (Xu et al., 2020)

Gambar pelatihan *LayoutLM* ini menjelaskan alur kerja pemrosesan data dalam Model *LayoutLM*. Pelatihan Model *LayoutLM* bekerja dengan cara membagi pemrosesan data menjadi dua tahapan. Tahap pertama adalah pemrosesan pada *text* dan *bounding box* yang merupakan posisi dari *text* tersebut. Tahap pertama dimulai dengan menggunakan *pre-build OCR parser* untuk mengekstrak informasi *text* beserta posisi *text* tersebut dalam bentuk *embedding*. Kedua informasi tersebut selanjutnya akan dilatih dengan menggunakan *pretrained LayoutLM*. Tahap kedua adalah dengan mengambil *Region of Interest (ROI)* dari *OCR parser* yang sama, kemudian gambar akan diproses dengan menggunakan *Faster R-CNN* dengan layer akhir berupa *Fully Connected Layers*. Hasil dari proses pada kedua tahapan ini berupa *embedding* yang akan dibandingkan untuk menjadi hasil akhir dari Model ini.

2.7 Metriks Evaluasi Model

Evaluasi pada Model dilakukan menggunakan *confussion matrix* serta nilai akurasi, presisi, *recall*, dan *f1-score* yang didapatkan Model pada data uji. *Confussion matrix* adalah suatu tabel yang dapat memberikan ringkasan tentang performa dari klasifikasi yang berhubungan dengan pengujian yang dilakukan (Ting, 2010). Penggunaan *confussion matrix* dapat menunjukkan hasil prediksi label dari setiap

kelas dengan lebih baik dengan memberikan nilai *true positive* (TP), *false positive* (FP), *false negatif* (FN) dan *true negative* (TN) yang merepresentasikan hasil prediksi Model (Ramsay et al., 2011).

Tabel 2.2 Confusion Matrix

	Prediksi Negatif	Prediksi Positif
Aktual Negatif	TN	FP
Aktual Positif	FN	TP

Tabel 2.1 merupakan tabel contoh penggunaan *confusion matrix*. *Confusion matrix* sering digunakan merepresentasikan hasil prediksi sebuah Model dalam permasalahan klasifikasi. Penjelasan dari istilah-istilah yang terdapat pada *confusion matrix* adalah sebagai berikut.

- True Positive* (TP) merupakan label positif yang berhasil diprediksi positif
- False Positive* (FP) merupakan label negatif yang salah diprediksi sebagai positif
- False Negative* (FN) merupakan label positif yang salah diprediksi negatif
- True Negative* (TN) merupakan label negatif yang berhasil diprediksi negatif

Hasil dari *confussion matrix* dapat digunakan untuk menghitung nilai akurasi, presisi, *recall*, dan *f1-score* dari Model yang telah dibuat. Perhitungan nilai ini dilakukan pada setiap label yang ada dalam proses pelatihan untuk mengetahui performa Model pada masing-masing label.

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.1)$$

Akurasi dihitung dengan menjumlahkan nilai TP dan TN pada hasil *confusion matrix* dan membaginya dengan nilai total data yang diprediksi. Hasil akurasi menunjukkan keakuratan model dalam memprediksi kelas yang tepat.

$$\text{Presisi} = \frac{TP}{TP + FP} \quad (2.2)$$

Presisi dihitung dengan membagi nilai TP dengan jumlah TP dan FP pada hasil *confusion matrix*. Hasil presisi keakuratan prediksi model dalam memprediksi data positif.

$$Recall = \frac{TP}{TP + FN} \quad (2.3)$$

Recall dihitung dengan membagi nilai TP dengan jumlah TP dan FN pada hasil *confusion matrix*. Hasil *recall* menunjukkan keberhasilan model dalam menemukan data berlabel positif.

$$F1\ Score = \frac{2 * (Recall * Precission)}{(Recall + Precission)} \quad (2.4)$$

F1-score dihitung dengan mencari nilai *recall* dan presisi terlebih dahulu. Penggunaan *f1-score* sering dilakukan pada *dataset* yang memiliki kelas yang tidak seimbang sebagai alternatif dari nilai akurasi.

BAB III

METODOLOGI PENELITIAN

Bab III merupakan metodologi penelitian tentang *finetuning* pada Model LayoutLM yang digunakan membaca nota pada penelitian ini. Bab ini membahas tentang tempat dan waktu penelitian, data yang digunakan, gambaran umum sistem, alur aplikasi serta perancangan sistem.

3.1 Tempat dan Waktu Penelitian

Pembuatan Tugas Akhir sistem pembacaan nota ini dilakukan di Kampus Teknologi Informasi Universitas Udayana. Waktu pelaksanaan pembuatan Tugas Akhir dimulai pada bulan September 2022 hingga pertengahan April 2023.

3.2 Data Penelitian

Data pelatihan yang digunakan dalam penelitian ini menggunakan data primer yang ditambahkan dengan data sekunder. Data primer merupakan data yang dikumpulkan oleh penulis untuk melakukan penelitian ini. Sedangkan data sekunder merupakan data yang tersedia secara publik dan dapat diambil untuk membantu penelitian.

3.2.1 Data Primer

Data Primer yang digunakan dalam *Finetuning* Model LayoutLM ini merupakan data yang dikumpulkan dari hasil bertransaksi pada berbagai restoran dan minimarket yang memberikan nota belanja *print out* digital. Total dari nota belanja yang telah dikumpulkan berjumlah 100. Nota-nota belanja ini selanjutnya difoto untuk dijadikan gambar yang digunakan sebagai data pelatihan Model LayoutLM.

3.2.2 Data Sekunder

Data Sekunder yang digunakan pada penelitian ini adalah *dataset WildReceipt* yang merupakan *dataset* nota belanja yang telah di-*scan*. Informasi yang terdapat pada *dataset WildReceipt* ini adalah informasi setiap kata, letak kata *bounding box*, serta label untuk setiap kata yang ada dalam nota belanja. *Dataset WildReceipt* ini memiliki data berjumlah 1267 data latih 472 data evaluasi (Theivaprakasham, 2022).

3.3 Instrumen Pembuatan Sistem

Instrumen pembuatan sistem memuat perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini. Pembuatan sistem memerlukan sebuah laptop untuk men-*deploy* sistem serta sebuah *smartphone* Android sebagai sarana pengujian aplikasi. Spesifikasi yang dimiliki oleh dari kedua perangkat tersebut adalah sebagai berikut.

Tabel 3.1 Spesifikasi Perangkat Keras

NO	Perangkat	Spesifikasi
1	Laptop	Windows, Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz 2.59 GHz, 24GB RAM, 512GB PCIe NVMe M.2 SSD, NVIDIA GeForce GTX 1660 TI
2	Smartphone Android	Android Oreo, API 28, ram 4gb

Tabel 3.1 memuat instrumen perangkat keras yang digunakan dalam pembuatan sistem pembacaan nota dengan OCR dan LayoutLM. Perangkat laptop digunakan dalam perancangan sistem serta sebagai *web server* yang menerima dan mengolah gambar. Perangkat Android berperan dalam pengujian aplikasi.

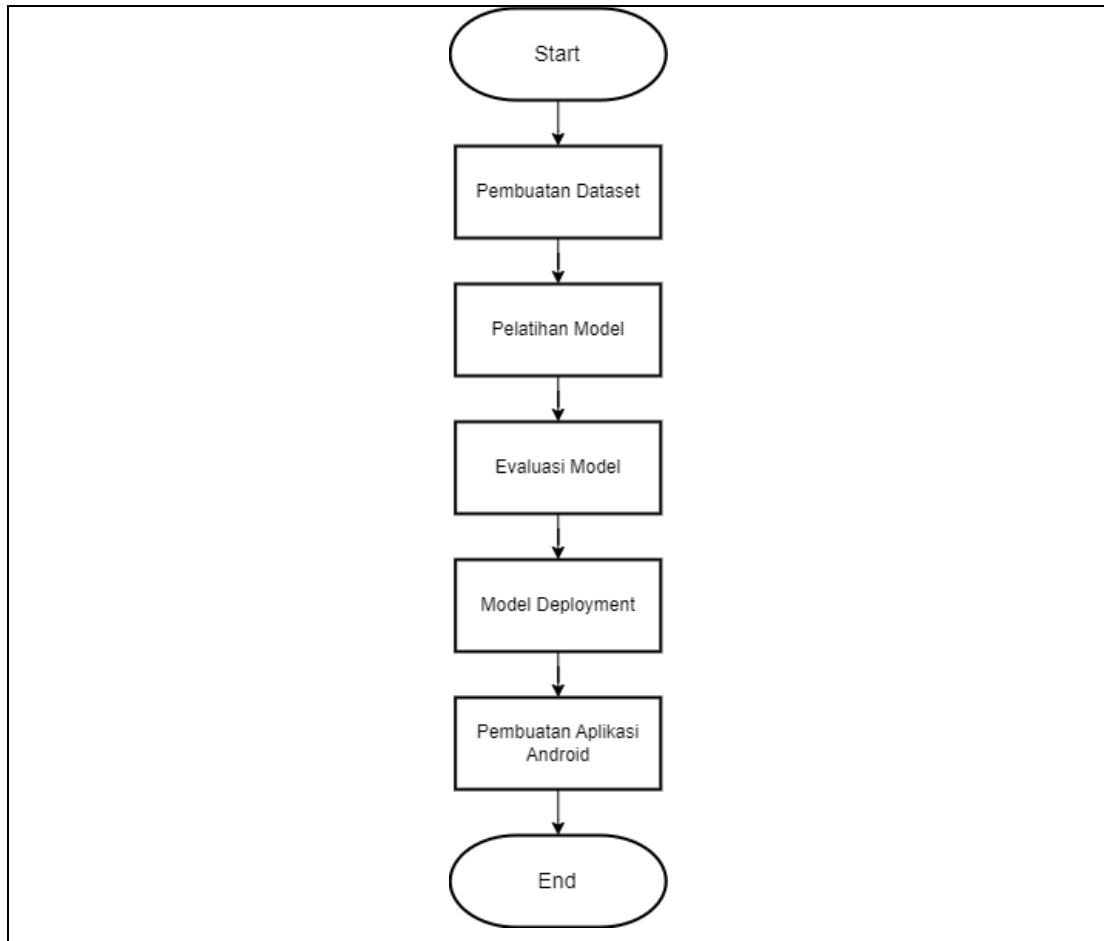
Tabel 3.2 Spesifikasi Perangkat Lunak

NO	Nama Software	Spesifikasi
1	Python	Versi 3.1.10
2	Android Studio	Versi 2021.2.1 patch 1
3	Flask	Versi 2.2.3

Tabel 3.2 memuat instrumen perangkat lunak yang digunakan dalam pembuatan sistem pembacaan nota dengan OCR dan LayoutLM. Python digunakan untuk pembuatan hingga *deployment* dari Model LayoutLM. Android Studio digunakan untuk membuat aplikasi dalam pengujian sistem. Flask digunakan dalam pembuatan *web server* sebagai sarana *deployment* dari Model.

3.4 Alur Penelitian

Alur penelitian membahas tentang proses pembuatan sistem *Finetuning* Model LayoutLM yang digunakan untuk membaca nota belanja. Proses yang dilakukan dalam pembuatan sistem ini meliputi pembuatan *dataset*, *Finetuning* Model LayoutLM, evaluasi Model, serta pembuatan aplikasi Android.

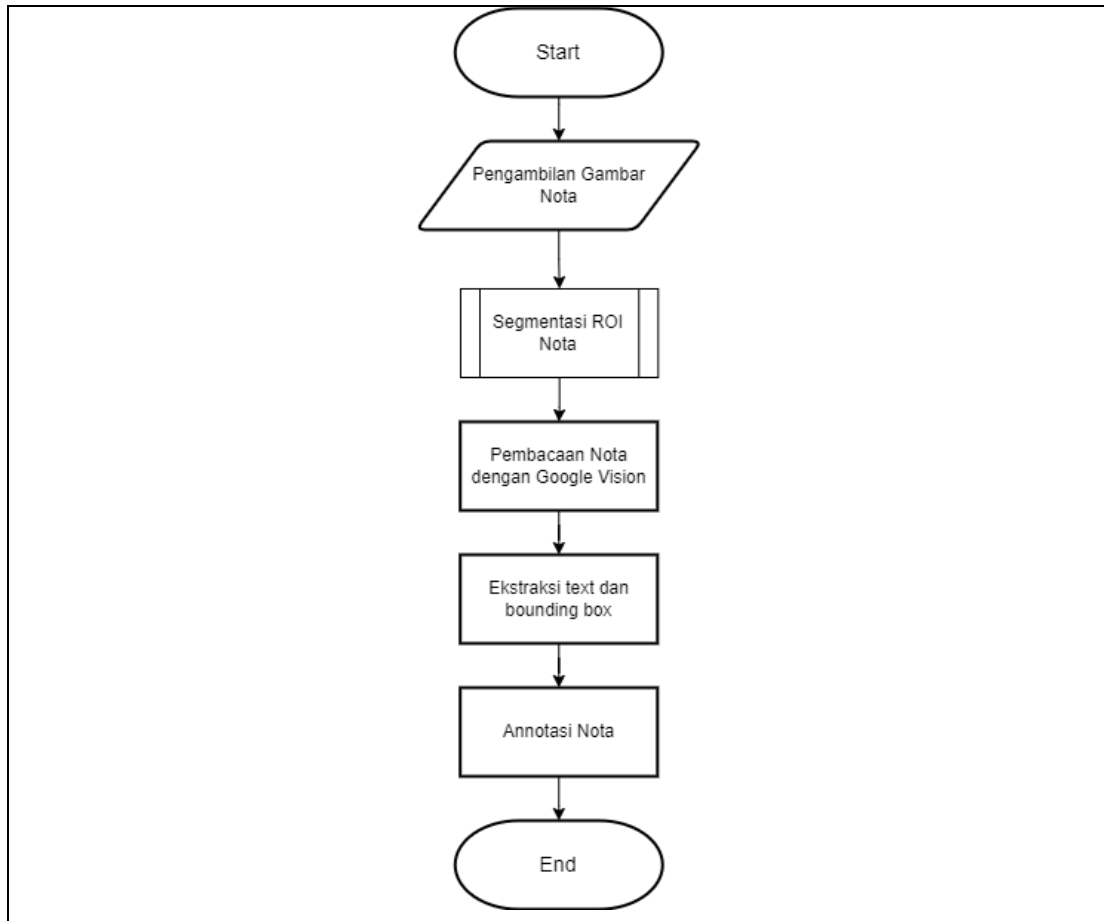


Gambar 3.1 *Flowchart Alur Penelitian*

Gambar 3.1 merupakan alur dari pengerjaan penelitian ini yang menjelaskan proses-proses yang dilakukan dalam pembuatan sistem pembacaan nota belanja. Pengerjaan dimulai dari pembuatan *dataset* hingga menjadi sebuah aplikasi yang dapat digunakan pada *smartphone* Android.

3.4.1 Pembuatan Dataset

Pembuatan *dataset* primer dilakukan dengan cara mengumpulkan nota belanja dari berbagai minimarket dan restoran. Nota belanja tersebut kemudian difoto dan diproses agar dapat digunakan dalam pelatihan Model.



Gambar 3.2 Flowchart Pembuatan Dataset

Gambar 3.2 menjelaskan tentang alur proses pembuatan *dataset* primer dalam penelitian ini. Proses pembuatan *dataset* ini menghasilkan sebuah file *.json* yang memuat informasi setiap kata, *bounding box*, dan label yang terdapat pada sebuah nota.

3.4.1.2 Segmentasi ROI

Proses segmentasi *region of interest* (ROI) diawali dengan pengubahan gambar menjadi citra *grayscale* dan menerapkan *Gaussian Blur* pada gambar nota. Proses selanjutnya adalah menerapkan *dilasi* pada gambar yang sudah di-*blur* agar tulisan yang terdapat pada nota tidak terbaca sebagai tepi yang akan disegmentasi.

Gambar hasil *dilasi* kemudian akan dideteksi dengan metode *Canny* untuk mendapatkan garis tepi nota. Berikut adalah gambaran dari proses segmentasi ROI ini.



Gambar 3.3 Proses Segmentasi ROI Nota

Gambar 3.3 memuat tentang proses segmentasi *ROI* pada gambar nota untuk mendapatkan gambar nota yang menyerupai hasil *scan*. Gambar ini memperlihatkan ilustrasi dari gambar asli, gambar hasil proses *grayscale*, gambar hasil *blurring* dan *dilasi*, serta gambar hasil deteksi tepi *Canny*.

Garis hasil deteksi tepi *Canny* digunakan sebagai dasar untuk memisahkan gambar nota dari *background* sehingga gambar hasil proses Segmentasi ROI ini menyerupai gambar hasil *scan*. Berikut adalah contoh gambar setelah melewati proses segmentasi ROI ini.

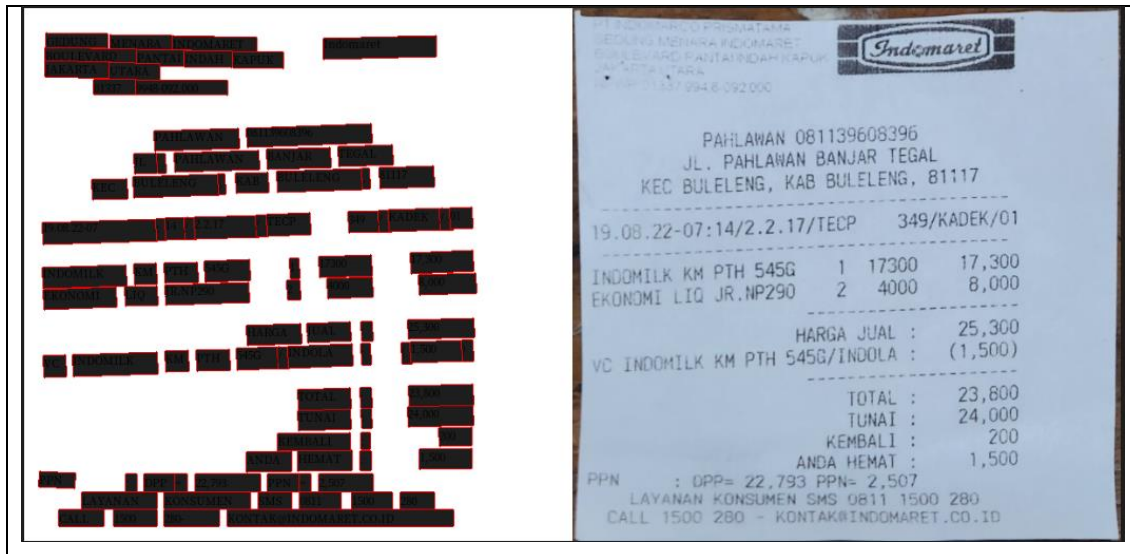


Gambar 3.4 Gambar Hasil Proses ROI

Gambar 3.4 merupakan contoh hasil dari nota setelah mengalami proses segmentasi ROI. Gambar ini kemudian akan di-*resize* hingga memiliki ukuran maksimal 1000x1000 *pixel*.

3.4.1.3 Pembacaan Google Vision

Pembacaan *Google Vision* dilakukan dengan menggunakan bantuan *library Layout Parser*. Proses ini bertujuan untuk melakukan segmentasi pada setiap kata yang ada di dalam nota, serta mendapatkan hasil pembacaan OCR dari setiap kata tersebut. Berikut adalah ilustrasi dari hasil segmentasi kata dengan *Google Vision* dan *Layout Parser*.



Gambar 3.5 Gambar Hasil Pembacaan Google Vision

Gambar 3.5 merupakan gambar hasil proses pembacaan *Google Vision* dengan menggunakan *library Layout Parser*. Masing-masing kata dan posisi kata yang telah terdeteksi kemudian akan disimpan sementara sebelum diproses kembali dalam proses *annotasi*.

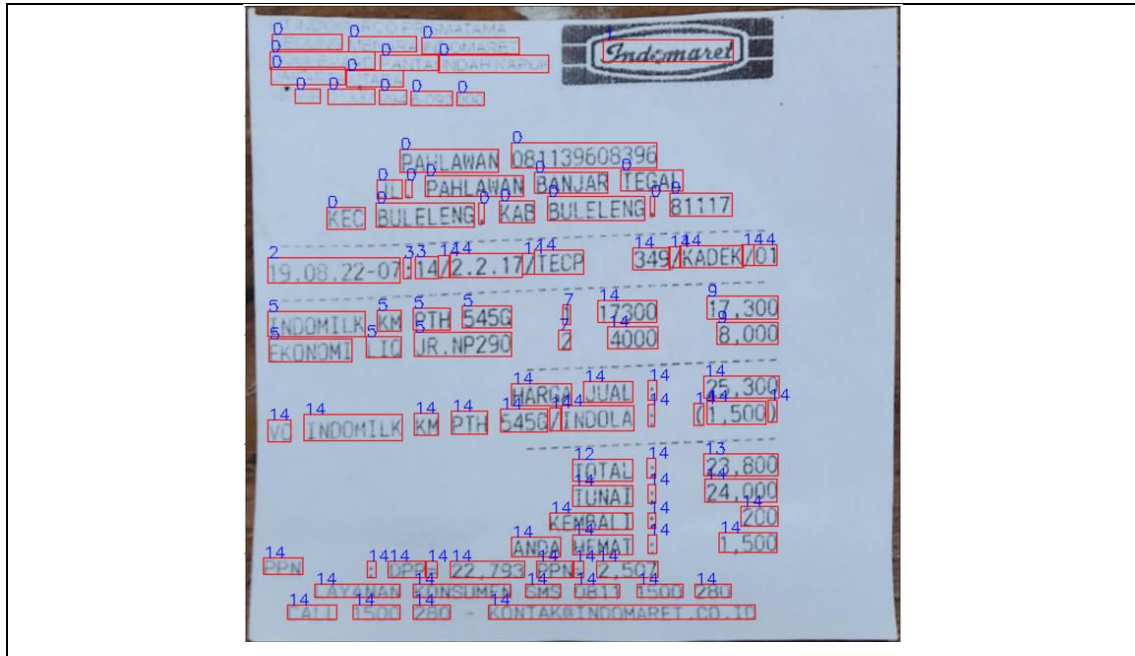
3.4.1.4 Annotasi Dataset

Proses *annotasi* merupakan proses pelabelan *dataset* primer yang telah dibaca oleh Google Vision. Proses ini dilakukan secara manual dengan cara memberikan nomor label pada setiap kata yang telah terdeteksi. Berikut adalah tabel yang digunakan untuk memberikan label pada setiap kata serta contoh gambar nota yang telah di-*annotasi*.

Tabel 3.3 Pelabelan Dataset

Nama Label	Kode Label
<i>Ignore</i>	0
<i>Store_name_value</i>	1
<i>Date_value</i>	2
<i>Time_value</i>	3
<i>Prod_item_key</i>	4
<i>Prod_item_value</i>	5
<i>Prod_quantity_key</i>	6
<i>Prod_quantity_value</i>	7
<i>Prod_price_key</i>	8
<i>Prod_price_value</i>	9
<i>Subtotal_key</i>	10
<i>Subtotal_value</i>	11
<i>Total_key</i>	12
<i>Total_value</i>	13
<i>Others</i>	14

Tabel 3.3 adalah daftar label yang digunakan dalam pembuatan *dataset* primer. Terdapat 15 label yang digunakan pada pembuatan *dataset*. Label-label yang digunakan pada hasil adalah *Store_name_value*, *Date_value*, *Time_value*, *Prod_item_value*, *Prod_quantity_value*, *Prod_price_value*, *Subtotal_value*, dan *Total_value*, dengan label lainnya digunakan sebagai label pembantu pemahaman Model.



Gambar 3.6 Gambar Hasil Proses Annotasi Dataset

Gambar 3.6 merupakan gambar hasil proses *annotasi dataset*. Gambar ini memperlihatkan salah satu data primer yang dibuat untuk melakukan *finetuning* pada Model LayoutLM. Setiap kata pada nota sudah dideteksi secara OCR, memiliki label serta mempunyai *bounding box* yang akan digunakan oleh Model LayoutLM untuk memahami informasi yang terkandung dalam nota.

Pembuatan *dataset* primer ini menghasilkan 100 nota berbahasa Indonesia yang berasal dari berbagai minimarket dan restoran. Tabel berikut adalah tabel sumber nota beserta jumlah sampel yang berasal dari sumber nota tersebut.

Tabel 3.4 Daftar Sampel *Dataset* Primer

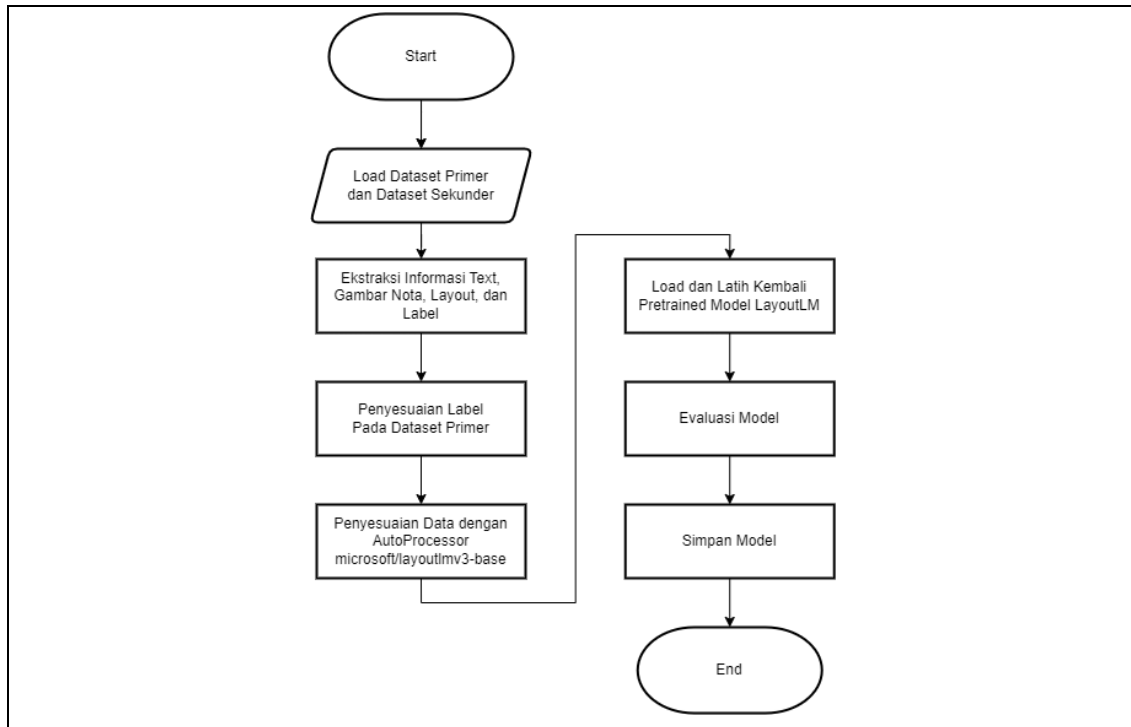
NO	Nama Toko	Total Sampel
1	Ayam Goreng Prambanan	1
2	Starbucks	1
3	Twisterdog	2
4	Indomaret	44
5	Out School Store	2
6	Joy Victoria	1
7	Wulan Busana	1
8	Alfamart	6
9	Ramen Ya	1
10	Mie Gacoan	2

11	Kusuma Makmur	2
12	Dewa Dewi Collection	1
13	SAS mart	2
14	Mixue	6
15	Komugi Bakery	1
16	Mc Donald	1
17	TIJE mart	1
18	UD Manik Galih	1
19	Circle K	11
20	TOOSI	7
21	Apotek Anugrah	1
22	Puri Sosis	1
23	Bakso Solo	1
24	Toko Frozen	1
25	Warung Bakso	1
26	Apotik Sumber Farma	1
TOTAL		100

Tabel 3.4 merupakan daftar nota yang digunakan sebagai data primer dalam penelitian ini. Data ini selanjutnya akan dibagi menjadi 80 data latih dan 20 data uji dalam pelatihan Model LayoutLM.

3.4.2 Finetuning Model LayoutLM

Model *Finetuning* adalah sebuah istilah dalam *machine learning* untuk menyelesaikan permasalahan *Natural Language Processing* (NLP). *Finetuning* merupakan proses yang menyerupai *transfer learning* pada *Convolutional Model*, di mana arsitektur model yang sama dapat digunakan untuk menyelesaikan berbagai permasalahan. *Finetuning* Model LayoutLM dilakukan dengan melatih model dasar dengan data baru yaitu nota yang telah dikumpulkan.



Gambar 3.7 Flowchart Pelatihan LayoutLM

Gambar 3.4 merupakan *flowchart* dari proses pelatihan Model LayoutLM. Alur pelatihan dimulai dengan memuat *dataset* dan melakukan ekstraksi informasi *layout* dan label dari setiap kata dalam nota. Informasi ini selanjutnya akan dimuat ke dalam proses *auto encoder* yang dimiliki oleh Model LayoutLM untuk menyamakan format data dengan format *original* pelatihan. Data *encoding* hasil pengubahan format ini digunakan untuk melakukan *Finetuning* pada Model LayoutLM. Hasil terbaik dari proses pelatihan Model ini disimpan untuk digunakan dalam mendeteksi nota.

3.4.3 Evaluasi Model

Evaluasi Model *finetuned* LayoutLM dilakukan dengan cara membandingkan hasil prediksi label dari setiap kata, dengan label aslinya. Evaluasi dan pengukuran akurasi Model dilakukan dengan menggunakan *confussion matrix* serta *classification report* yang menunjukkan hasil pada setiap label yang berbeda. Selain itu, pengujian performa model juga dilakukan secara *realtime* dalam beberapa kasus pengujian yang berbeda.

3.4.3.1 Confussion Matrix

Tahap penilaian *confussion matrix* memperlihatkan hasil deteksi pada setiap label yang dapat dideteksi dengan melihat jumlah label yang berhasil dideteksi dengan benar pada diagonal utama hasil dari *confussion matrix*. Data yang digunakan memiliki 9 label yang harus dibedakan oleh Model antara lain label *Store_name_value*, *Date_value*, *Time_value*, *Prod_item_value*, *Prod_quantity_value*, *Prod_price_value*, *Total_key*, *Total_value*, dan *Others*.

3.4.3.2 Classification Report

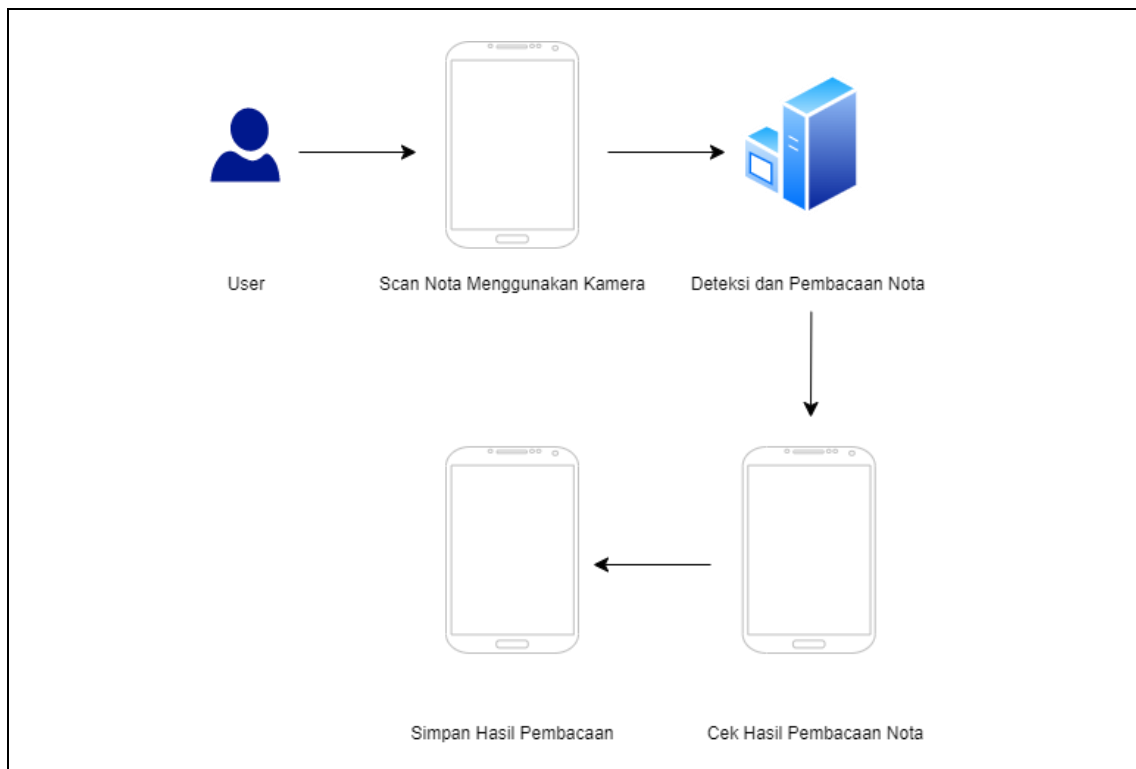
Tahap penilaian *classification report* memperlihatkan hasil nilai akurasi, presisi, dan *recall*. Penilaian *classification report* ini akan menilai 9 label yang merupakan informasi yang ingin diekstrak dari hasil pembacaan nota, yaitu *Store_name_value*, *Date_value*, *Time_value*, *Prod_item_value*, *Prod_quantity_value*, *Prod_price_value*, *Subtotal_value*, *Total_value*, dan *Others*.

3.4.4 Pengujian Sistem

Pengujian sistem dilakukan dengan cara men-*deploy* model yang telah dibuat ke dalam sebuah *web server*. Sebuah aplikasi Android akan mengirimkan gambar nota ke *web server* tersebut melalui *API Call* agar nota tersebut dapat dibaca model. Kemampuan sistem untuk membaca nota diuji dalam beberapa skenario pengujian yang terdiri dari pengujian nota yang beragam, pengujian variasi kecerahan nota, dan pengujian variasi panjang nota.

3.5 Gambaran Umum Sistem

Gambaran umum sistem menjelaskan tentang desain perancangan dari sistem yang digunakan pada pembuatan sistem pembacaan nota ini. Sistem yang dibuat terbagi menjadi rancangan server beserta rancangan aplikasi android sebagai berikut.

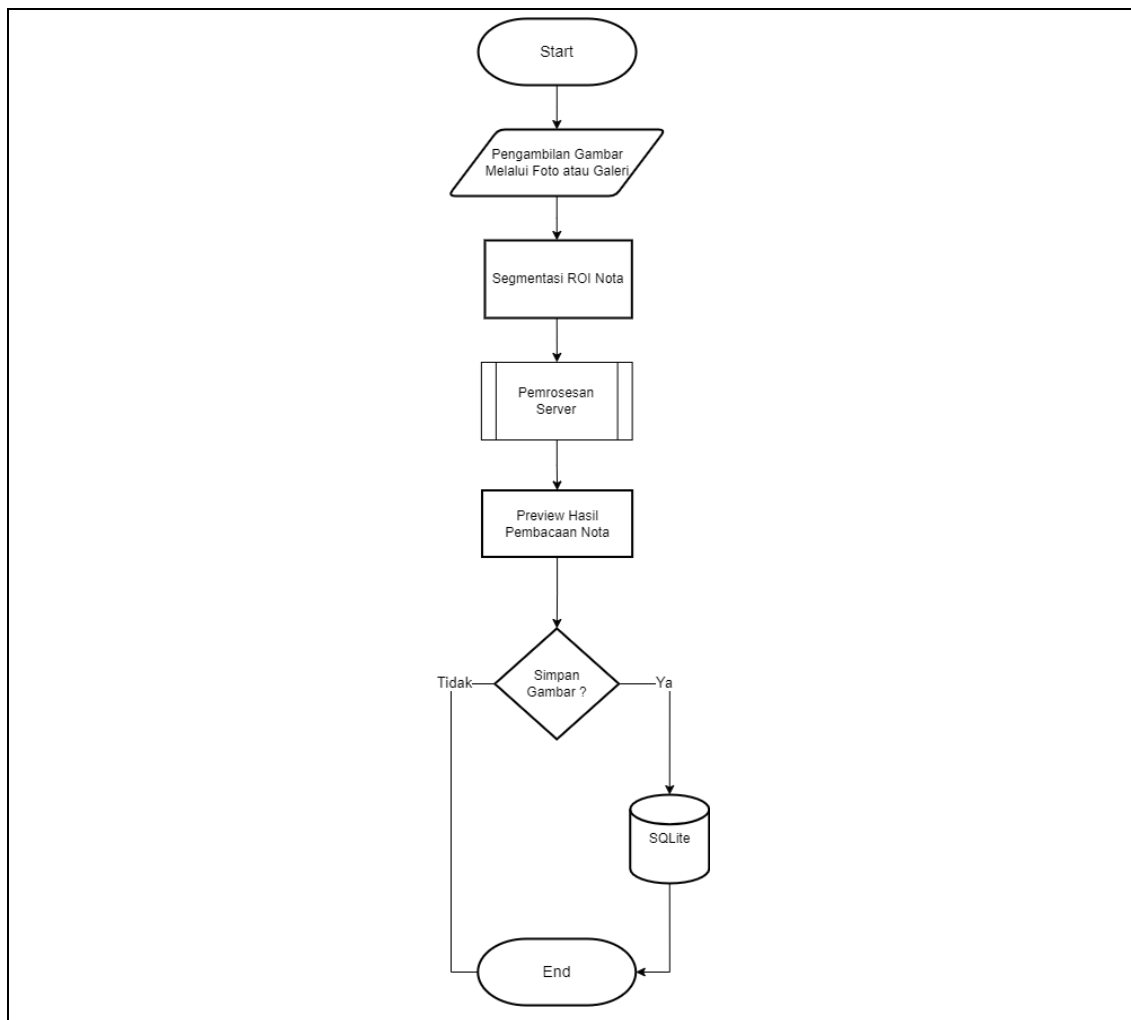


Gambar 3.8 Gambaran Umum Sistem

Gambar 3.5 merupakan gambaran umum dari aplikasi android yang telah dibuat. Aplikasi ini memiliki dua fungsi utama. Fungsi pertama adalah untuk mengirim foto dari nota belanja kepada *web server* dan menyimpan hasil pembacaan. Fungsi kedua dari aplikasi adalah untuk melihat *history* data belanja bulanan yang tersimpan pada sistem.

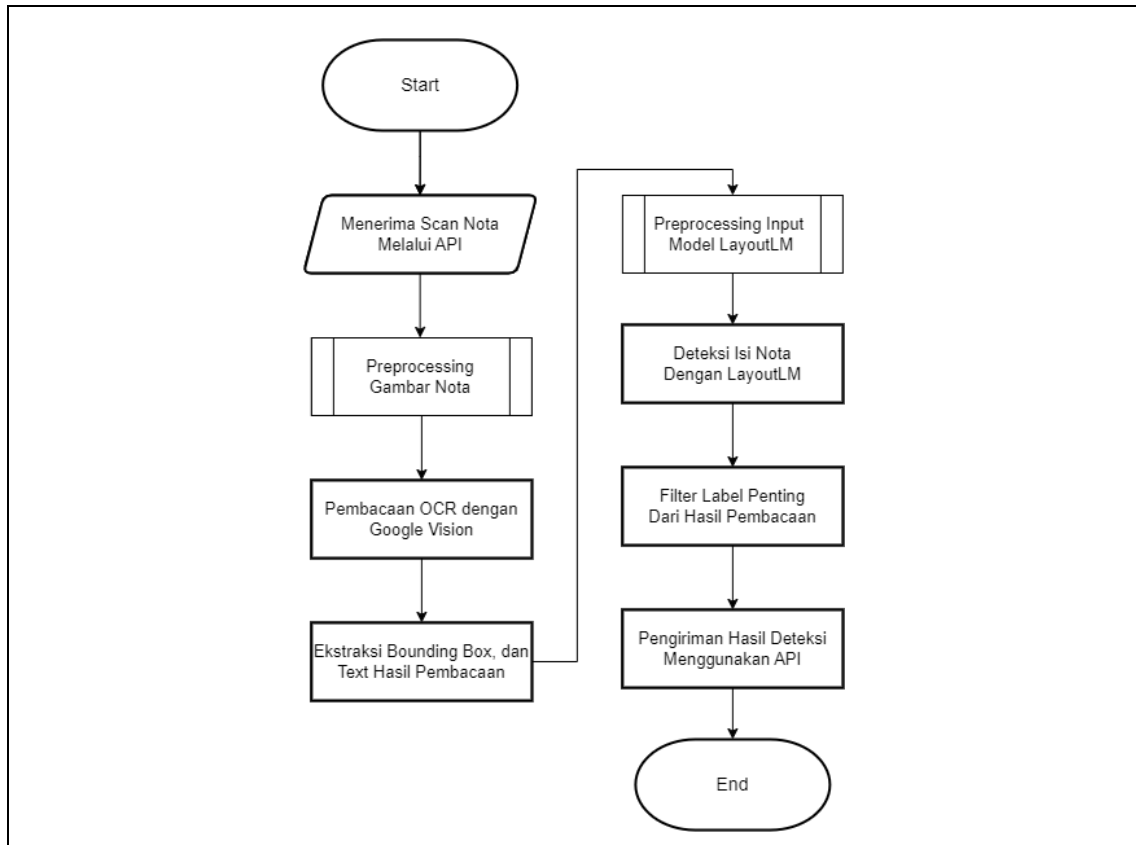
3.6 Alur Aplikasi

Alur aplikasi menjelaskan tentang alur pemrosesan data yang terdapat pada aplikasi pembacaan nota belanja ini. Alur aplikasi ini menjelaskan tentang kedua fungsi utama dari aplikasi Android serta interaksi aplikasi dengan *web server* dalam bentuk *flowchart*.



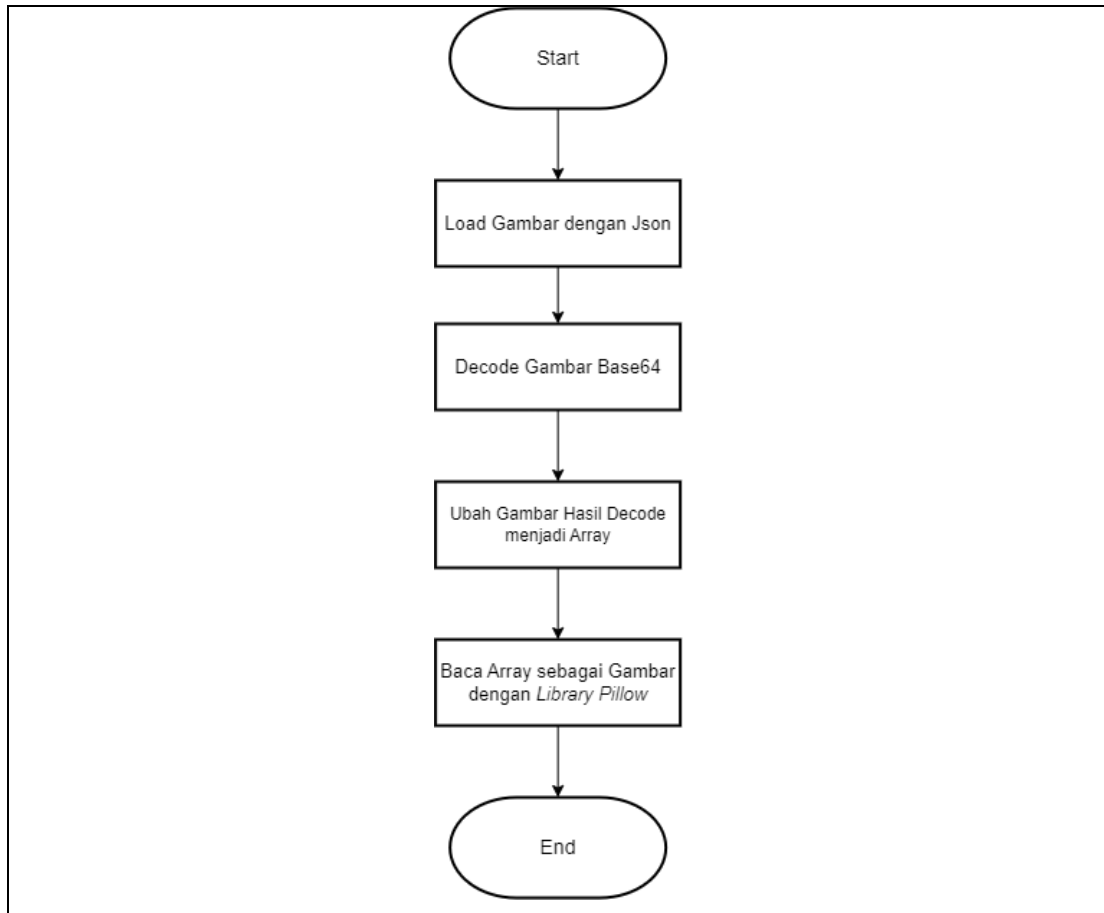
Gambar 3.9 Flowchart Pembacaan Nota pada Aplikasi Android

Gambar 3.6 merupakan *flowchart* alur pembacaan nota yang terdapat pada aplikasi Android. Pembacaan nota belanja dapat dilakukan dengan mengirimkan foto dari nota yang ingin dibaca kepada *web server*. Hasil dari pembacaan nota kemudian dikirim kembali oleh *web server* untuk dapat disimpan pada sistem Android.



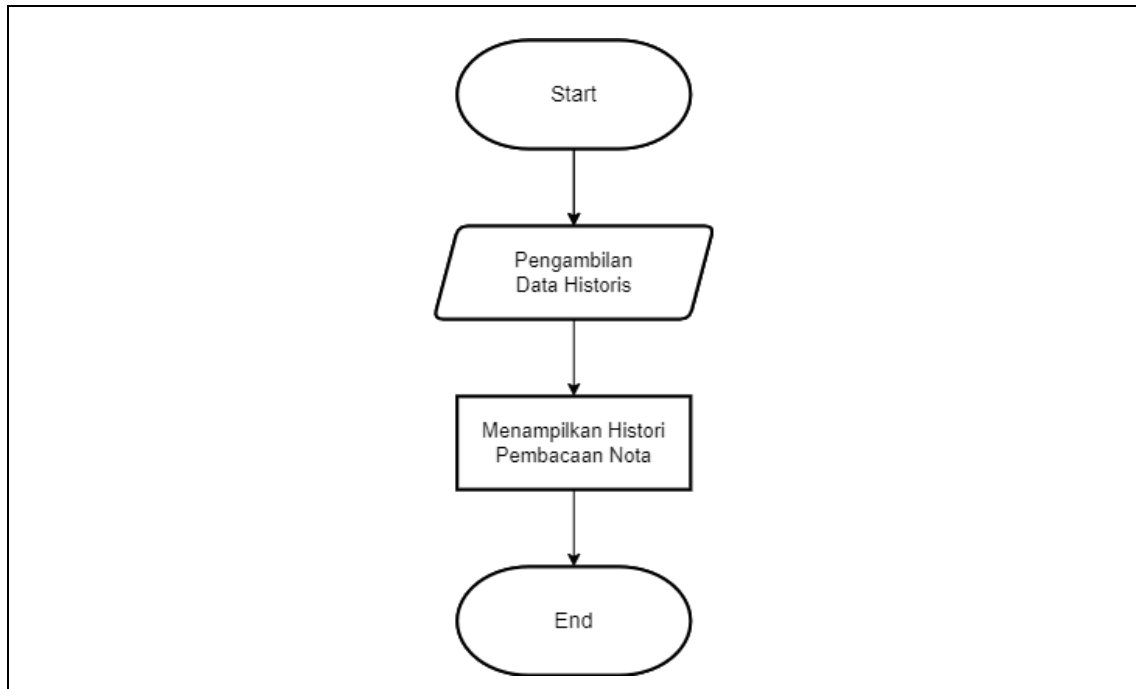
Gambar 3.10 Flowchart Proses Pembacaan Nota pada Web Server

Gambar 3.7 merupakan gambaran *flowchart* pemrosesan pada *web server* yang dijalankan dengan menggunakan *framework Flask*. *Web server* akan menerima gambar yang telah dikirimkan dari aplikasi Android dan melakukan *preprocessing* pada gambar tersebut agar dapat terbaca pada sistem. Gambar tersebut selanjutnya akan diproses kembali untuk mendapatkan *embedding text* dan *layout* nota serta hasil prediksi Model. Hasil prediksi dari Model kemudian diinterpretasikan menjadi informasi nama toko, tanggal dan waktu transaksi, produk yang dibeli, serta total belanja yang akan dikirim kembali kepada aplikasi android.



Gambar 3.11 Flowchart Preprocessing Gambar pada Server

Gambar 3.8 merupakan gambaran *flowchart* proses pemrosesan gambar pada *server*. Pertama gambar yang diterima melalui *API Call* akan dibaca menggunakan *library Json* untuk membaca *file enkripsi*. *File* enkripsi tersebut lalu diterjemahkan menggunakan *library Base64*. Hasil terjemahan *file* kemudian diproses untuk dijadikan sebuah *array* dan dibaca sebagai sebuah gambar dengan menggunakan *library Pillow*.

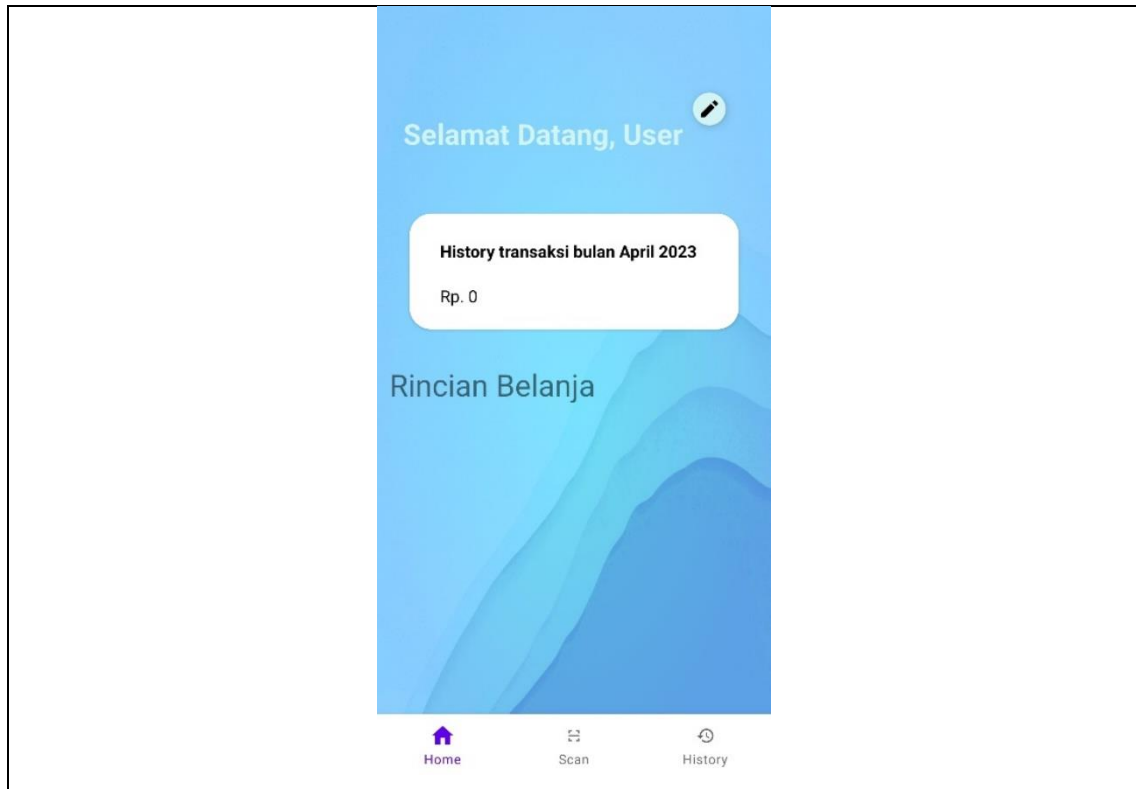


Gambar 3.12 *Flowchart Melihat Histori Pembacaan Nota pada Aplikasi Android*

Gambar 3.9 merupakan *flowchart* alur melihat riwayat pembacaan nota yang terdapat pada aplikasi Android. Proses melihat riwayat pembacaan nota diawali dengan pengambilan data historis dari *database SQLite* Android. Data historis kemudian ditampilkan pada aplikasi Android.

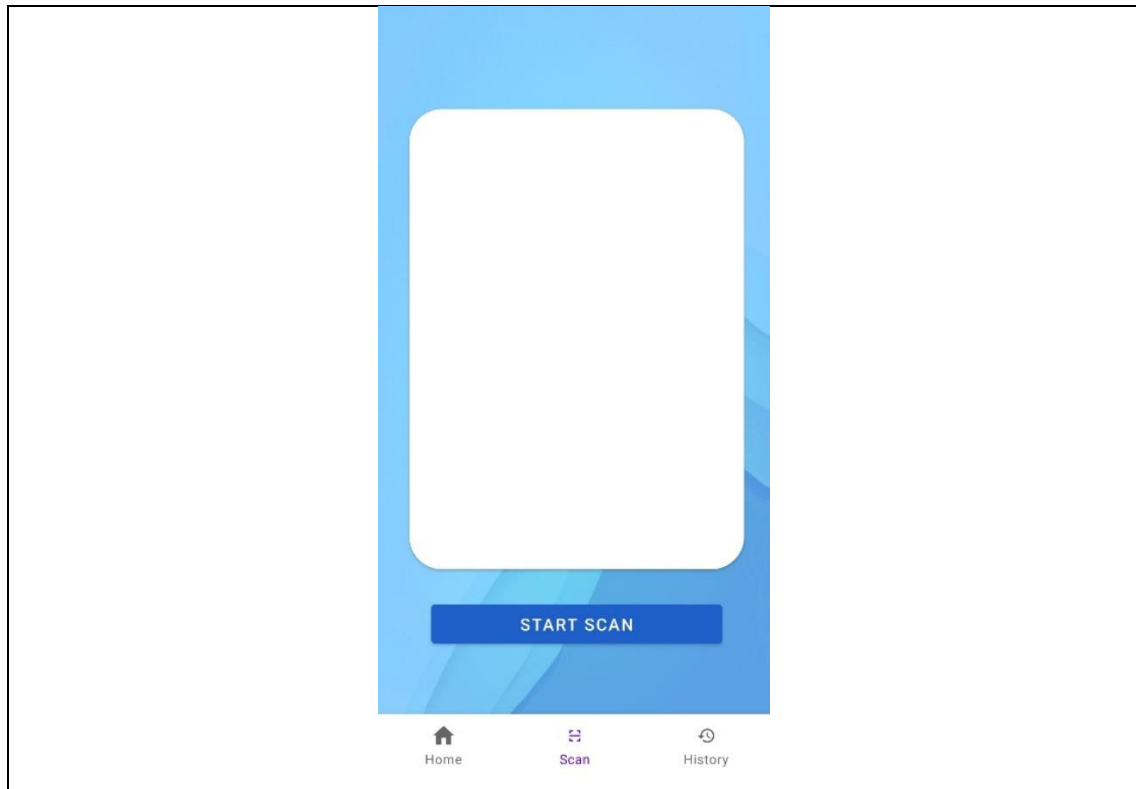
3.7 Rancangan Sistem

Rancangan sistem memuat tentang tampilan proses yang ada pada sistem Android. Rancangan dari sistem ini memuat proses pengambilan gambar, segmentasi nota belanja, tampilan hasil deteksi, serta halaman *history* yang memuat informasi riwayat pembacaan nota yang telah tersimpan.




Gambar 3.13 Rancangan Tampilan Depan Sistem

Gambar 3.10 adalah gambar rancangan tampilan depan saat aplikasi ini dibuka. Aplikasi akan menampilkan riwayat hasil *scan* terbaru, serta dua buah tombol untuk men-*scan* nota baru serta melihat riwayat hasil *scan* dengan lebih lengkap.



Gambar 3.14 *Rancangan Halaman Scan*

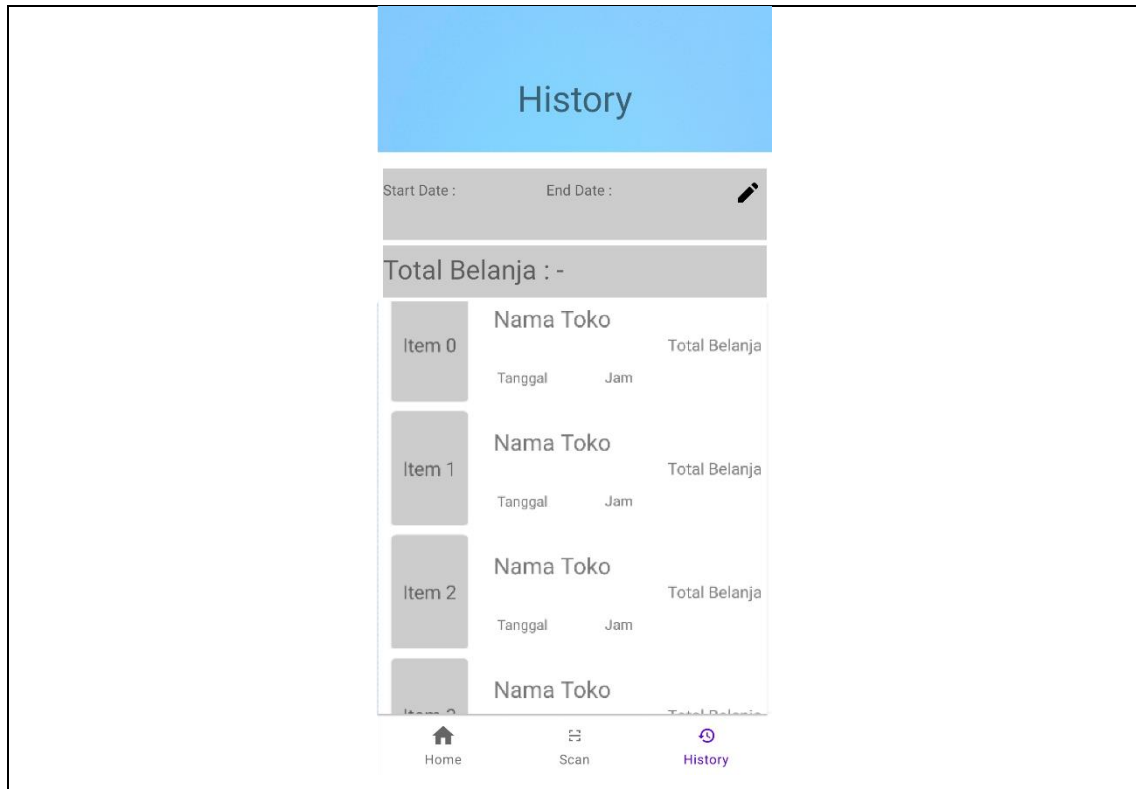
Gambar 3.11 adalah gambar rancangan halaman *scan* nota pada aplikasi Android. Gambar dari nota yang dipilih akan terlihat pada layar. Tombol *start scan* dapat ditekan untuk mengirimkan gambar nota ke *web server*.



The image shows a mobile application interface for a receipt preview. It features a blue border with a repeating pattern of stars and shopping bags. The main content area is white and contains a large rounded rectangle at the top, likely for a photo or logo. Below this are four input fields: 'Nama Toko', 'Tanggal', 'Waktu', and 'Total Belanja'. At the bottom, there is a 'Product Name' label, a percentage sign, the amount 'Rp. 150.000', and a shopping bag icon. At the very bottom are two buttons: a red 'Cancel' button and a green 'Save' button.

Gambar 3.15 Rancangan Hasil Preview Pembacaan

Gambar 3.12 adalah gambar rancangan halaman *preview* setelah nota dikirim dan diberikan hasil pembacaan oleh *web server*. Gambar akan menampilkan informasi-informasi penting yang akan disimpan oleh aplikasi. Halaman ini memiliki dua buah tombol navigasi. Tombol *cancel* digunakan untuk kembali ke halaman sebelumnya dan membatalkan *scan* serta tombol *save* dapat digunakan untuk menyimpan hasil *scan*.



Gambar 3.16 Rancangan Riwayat Hasil Scan

Gambar 3.13 adalah gambar rancangan halaman riwayat hasil *scan*. Terdapat rentang waktu yang dapat dipilih serta total pengeluaran yang akan di-*update* sesuai dengan rentang waktu yang dipilih. Data yang sesuai dengan rentang waktu akan terlihat pada daftar rincian yang memuat informasi nama toko, tanggal, serta total belanja pada transaksi tersebut.

BAB IV

HASIL DAN PEMBAHASAN

Bab IV hasil dan pembahasan membahas mengenai proses pengerjaan sistem yang digunakan dalam melakukan *finetuning* Model LayoutLM untuk membaca nota berbahasa Indonesia beserta hasil yang ditemukan selama pengerjaan sistem.

4.1 Pembuatan Dataset

Pembuatan *dataset* dilakukan dengan cara mengumpulkan nota belanja yang didapatkan setelah bertransaksi di toko Alfamart, Circle K, Indomaret, Mixue, dll. Nota-nota tersebut selanjutnya difoto menggunakan perangkat *smartphone*.

4.1.1 Segmentasi Gambar

Proses segmentasi dilakukan pada gambar nota yang telah dikumpulkan untuk menghilangkan *background*. Hasil dari proses segmentasi mendapatkan sebuah gambar yang berfokus kepada nota yang merupakan informasi utama dalam pengerjaan sistem.

```
file_name = '20221013_192759.jpg'
file_path = os.path.join(DIR, 'Nota', file_name)
img_read = cv2.imread(file_path)
img_read = cv2.cvtColor(img_read, cv2.COLOR_BGR2RGB)
if img_read is None:
    raise Exception(f"Image {file_name} not found")

resize_ratio = 1000 / img_read.shape[0]
img_rezise = resize_img(img_read, resize_ratio)
gray = cv2.cvtColor(img_rezise, cv2.COLOR_BGR2GRAY)
plt.imshow(img_rezise)
plt.show()
```

Kode Program 4.1 Pembacaan File Gambar Nota

Kode Program 4.1 merupakan kode program yang digunakan untuk membaca *file* foto dengan menggunakan *library OpenCv*. Proses pertama dalam mengolah gambar adalah proses `resize_img` untuk memperkecil ukuran gambar dan operasi `cv2.cvtColor` untuk mengambil gambar dalam bentuk *grayscale*.



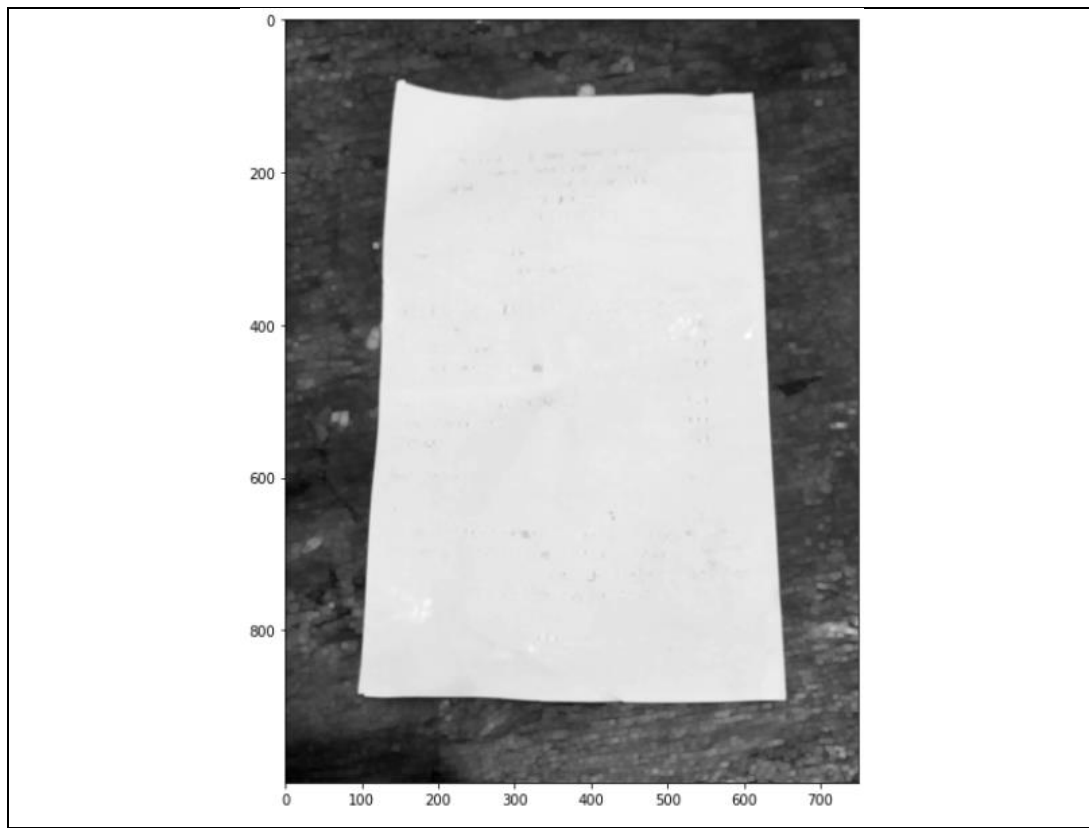
Gambar 4.1 Hasil Proses Pembacaan Gambar Serta Resize

Gambar 4.1 adalah hasil dari proses pembacaan gambar dan proses `resize_img` yang telah dilakukan. Gambar nota hasil memiliki ukuran panjang 1000. Selain itu sebuah gambar *grayscale* disimpan dalam variabel `gray`.

```
blurred = cv2.GaussianBlur(gray, (3, 3), 3)
rectKernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5, 5))
dilated = cv2.dilate(blurred, rectKernel)
plt.figure(figsize=(10,10))
plt.imshow(dilated, cmap='gray')
plt.show()
```

Kode Program 4.2 Preprocessing Nota Tahap 1

Kode Program 4.2 adalah kode program untuk memproses gambar *grayscale* dengan menggunakan proses *blurring*, dan proses dilasi. Proses *blurring* dilakukan dengan menggunakan `GaussianBlur` pada gambar *grayscale* dan proses `dilate` pada gambar yang telah di-*blur*.



Gambar 4.2 Hasil Proses Blurring dan Dilasi Pada Gambar

Gambar 4.2 adalah hasil dari proses *blurring* dan dilasi pada gambar *grayscale*. Proses *blurring* dan proses dilasi dilakukan pada gambar agar tulisan pada gambar menjadi tidak terbaca saat dilakukan deteksi tepi.

```
def auto_canny(image, sigma=1):
    # compute the median of the single channel pixel intensities
    v = np.median(image)

    # apply automatic Canny edge detection using the computed
    median
    lower = int(max(0, (1.0 - sigma) * v))
    upper = int(min(255, (1.0 + sigma) * v))
    edged = cv2.Canny(image, lower, upper)

    # return the edged image
    return edged
```

```

edged = auto_canny(dilated)

plt.figure(figsize=(10,10))

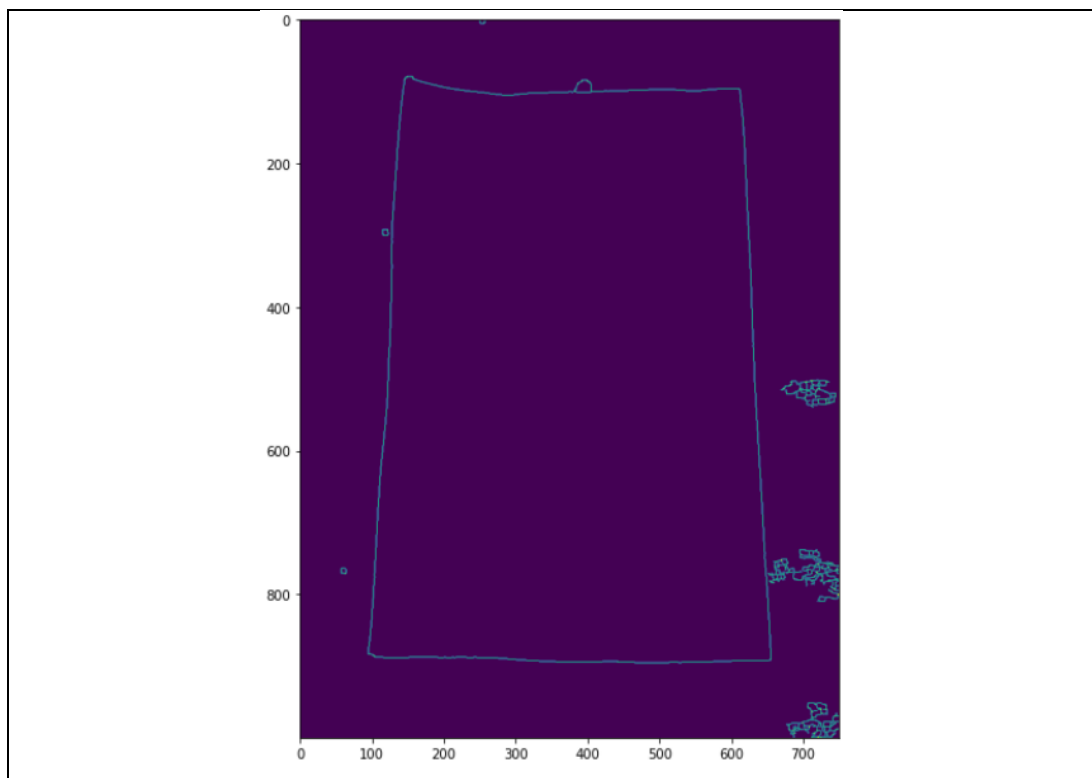
plt.imshow(edged)

plt.show()

```

Kode Program 4.3 *Proses Deteksi Tepi Canny*

Kode Program 4.3 merupakan kode program yang digunakan untuk mendeteksi tepi nota agar dapat dilakukan proses segmentasi. Hasil dari proses deteksi tepi ini dilanjutkan dengan mendeteksi area *contour* terbesar.



Gambar 4.3 *Hasil Deteksi Tepi Dengan Menggunakan Canny*

Gambar 4.3 merupakan hasil dari proses deteksi tepi pada nota yang sudah di-*blur*. Hasil dari proses deteksi ini adalah tepi dari nota dapat terlihat dan tulisan yang ada di dalam nota tidak terdeteksi sebagai tepi yang membuat proses segmentasi dapat berjalan dengan lebih baik.

```

detected_lines = cv2.HoughLinesP(edged, rho = 0.5, theta =
1*np.pi/180, threshold = 30, minLineLength = 20, maxLineGap = 100)
edged_line = np.zeros_like(edged)
line_extender = 0
for line in detected_lines:
    x1, y1, x2, y2 = line[0]

    is_vertical = abs(x1 - x2) < abs(y1 - y2)
    if is_vertical:
        x1 = int(x1-(abs(x1 - x2)/2*line_extender)*0.314)
        x2 = int(x2+(abs(x1 - x2)/2*line_extender)*0.314)

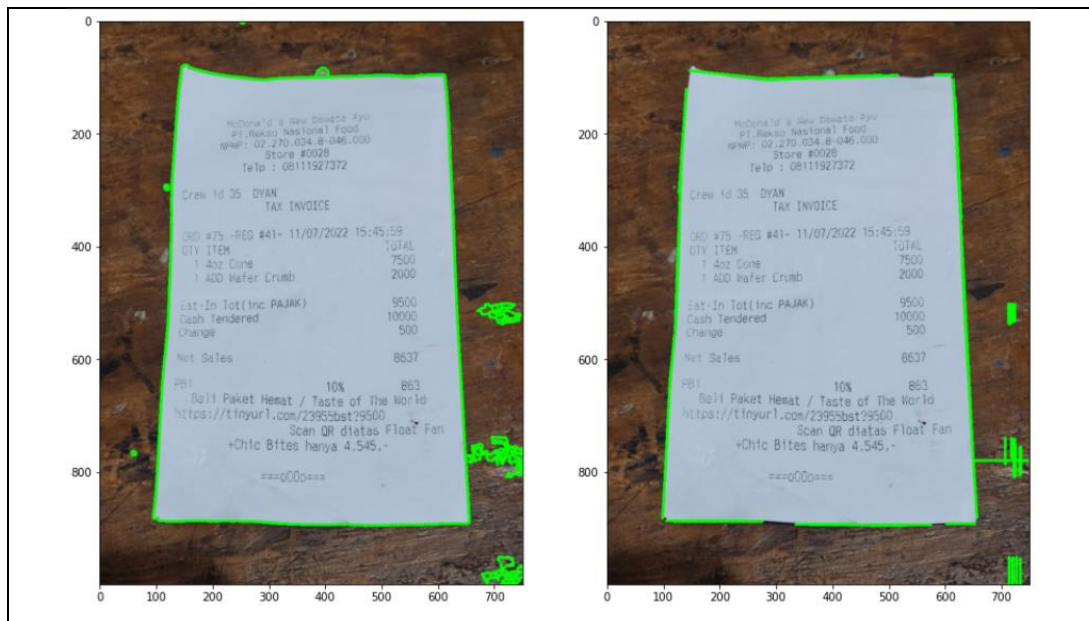
        if y1<y2:
            y1 = int(y1-(abs(y1 - y2)/2*line_extender))
            y2 = int(y2+(abs(y1 - y2)/2*line_extender))
        else:
            y1 = int(y1+(abs(y1 - y2)/2*line_extender))
            y2 = int(y2-(abs(y1 - y2)/2*line_extender))
    else:
        x1 = int(x1-(abs(x1 - x2)/2*line_extender))
        x2 = int(x2+(abs(x1 - x2)/2*line_extender))
        y1 = int(y1-(abs(y1 - y2)/2*line_extender)*0.314)
        y2 = int(y2+(abs(y1 - y2)/2*line_extender)*0.314)

    cv2.line(edged_line, (x1, y1), (x2, y2), (255), 1)
#main and backup method
contours_m, _ = cv2.findContours(edged, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
contours_b, hierarchy = cv2.findContours(edged_line,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
all_contours_m = cv2.drawContours(img_resize.copy(), contours_m, -
1, (0,255,0), 3)
all_contours_b = cv2.drawContours(img_resize.copy(), contours_b, -
1, (0,255,0), 3)
plt.figure(figsize=(15,15))
plt.subplot(1,2,1)
plt.imshow(all_contours_m)
plt.subplot(1,2,2)
plt.imshow(all_contours_b)
plt.show()

```

Kode Program 4.4 *Deteksi Kontur Hasil tepi Canny*

Kode Program 4.4 adalah kode program yang digunakan untuk mengekstrak tepi pada nota yang telah dideteksi tepinya oleh deteksi tepi *Canny*. Kode program ini menampilkan gambar asli dengan kontur tepi yang telah terdeteksi, serta membuat garis tambahan pada tepi dengan menggunakan fungsi `cv2.HoughLinesP`.



Gambar 4.4 Deteksi Kontur Hasil tepi *Canny*

Gambar 4.4 adalah gambar hasil deteksi kontur yang digambar pada gambar asli. Terdapat dua buah metode yang digunakan dalam penggambaran kontur. Metode pertama adalah dengan menggambarkan kontur secara langsung setelah deteksi tepi *Canny*. Metode kedua adalah dengan menggunakan fungsi `cv2.HoughLinesP` pada hasil *Canny* untuk mendeteksi garis terluar pada nota.

```
largest_contours = sorted(contours_m, key = cv2.contourArea,
reverse = True) [0:4]

image_with_largest_contours = cv2.drawContours(img_resize.copy(),
largest_contours, -1, (0,255,0), 3)

plt.figure(figsize=(10,10))
plt.imshow(image_with_largest_contours)
plt.show()
```

Kode Program 4.5 Pemilihan Kontur

Kode program 4.5 adalah kode program yang digunakan untuk pemilihan kontur terbesar yang digunakan untuk segmentasi nota. Kontur dengan area terbesar yang terdeteksi dengan fungsi `cv2.contourArea` akan terlihat pada gambar dan sisanya diabaikan menyesuaikan kategori pohon kontur.



Gambar 4.5 Hasil Pemilihan Kontur

Gambar 4.5 adalah gambar hasil pemilihan kontur pada nota. Kontur yang terbesar berada pada gambar sesuai dengan area dan hierarki pohon kontur yang dimiliki oleh kontur tersebut.

```
try:
    receipt_contour = get_receipt_contour(largest_contours)

    detected_receipt = cv2.drawContours(img_resize.copy(),
[receipt_contour], -1, (0, 255, 0), 2)

    result = wrap_perspective(img_resize.copy(),
contour_to_rect(receipt_contour, corner_tolerance = 0))

    print("using main method successfully")
except:

    print("using backup method")
    longest = 0

    for i, cont in enumerate(largest_contours):
        x, y, w, h = cv2.boundingRect(largest_contours[i])

        cur_length = w+h

        if (cur_length > longest):
```

```

        longest = cur_length
        biggest_idx = i

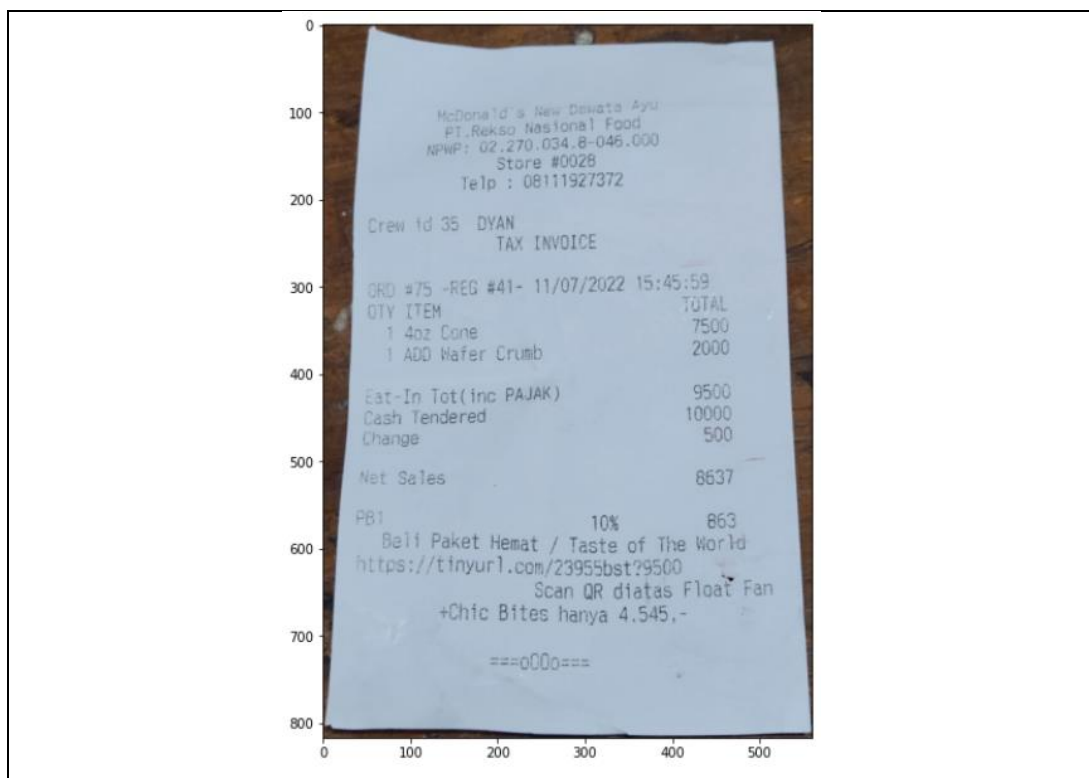
    x,y,w,h = cv2.boundingRect(largest_contours[biggest_idx])
    result = img_resize[y:y+h, x:x+w]

plt.figure(figsize=(10,10))
plt.imshow(result)
plt.show()

```

Kode Program 4.6 Segmentasi Nota

Kode program 4.6 adalah kode yang digunakan untuk melakukan segmentasi pada nota. Kode program ini menggabungkan tahap-tahap deteksi nota sebelumnya serta mengambil ordinat yang dimiliki oleh kontur terbesar untuk melakukan segmentasi.



Gambar 4.6 Nota Hasil Segmentasi

Gambar 4.6 adalah hasil setelah nota mengalami segmentasi. Gambar hasil segmentasi hanya berfokus pada nota agar pembacaan OCR dapat dilakukan dengan benar. Langkah selanjutnya adalah deteksi setiap kata pada gambar dengan menggunakan API *Google Vision*.

4.1.2 Pembacaan Karakter

Pembacaan karakter adalah tahap pembacaan setiap kata yang terdapat pada nota dengan menggunakan *Google Vision*. Proses pembacaan nota ini dibantu dengan *library Layout Parser*.

```
ocr_agent = lp.GCVAgent.with_credential(os.path.join(DIR, 'gcv_credential.json'), languages = ['id'])

res = ocr_agent.detect(image_result, return_response=True)
texts = ocr_agent.gather_text_annotations(res)
lp.draw_text(image_result, texts, font_size=12, with_box_on_text=True, text_box_width=3)
```

Kode Program 4.7 Pembacaan Karakter Dengan *Google Vision*

Kode program 4.7 adalah kode program yang digunakan untuk membaca karakter dengan *Google Vision*. Pertama-tama dilakukan inisiasi agen pembaca OCR pada *Layout Parser* dengan menggunakan kredensial dari API *Google Vision*. Lalu, dengan menggunakan fungsi *detect*, nota akan dibaca untuk menghasilkan *bounding box* serta kata yang terdapat pada nota.



Gambar 4.7 Hasil Deteksi *Google Vision*

Gambar 4.7 adalah gambar hasil pembacaan karakter oleh *Google Vision*. Gambar bagian kiri memperlihatkan penggambaran letak *bounding box* yang memuat sebuah *text* dari setiap kata yang terdeteksi oleh *Google Vision*. Pada bagian kanan merupakan gambar nota yang dikirimkan pada *Google Vision*.

```

import json
from tqdm.notebook import tqdm
receipt_list = {}

ocr_agent =
lp.GCVAgent.with_credential(os.path.join(DIR, 'gcv_credential.json'
), languages = ['id'])

for filename in tqdm(os.listdir('Nota_Segmented')):
    filepath = os.path.join(DIR, 'Nota_Segmented', filename)
    image_result = get_receipt(filepath)
    res = ocr_agent.detect(image_result, return_response=True)
    texts = ocr_agent.gather_text_annotations(res)

    inference_words = []
    for words_bbox in texts:
        inference_words.append(words_bbox.text)

    inference_boxes = []
    for words_bbox in texts:
        h = np.min(words_bbox.block.points, axis=0)
        w = np.max(words_bbox.block.points, axis=0)
        inference_boxes.append([h[0], h[1], w[0], w[1]])

    receipt_json = {}
    receipt_json['file_name'] = filename
    receipt_json['size'] = image_result.shape
    receipt_json['bboxes'] = [normalize_bbox(box,
image_result.shape) for box in inference_boxes]
    receipt_json['words'] = inference_words

    receipt_list[filename] = receipt_json

    with open(f"Annotation/{filename.split('.jpg')[0]}.json", "w")
as outfile:
    json.dump(receipt_list, outfile)

```

Kode Program 4.8 Kode Program Otomasi Pembacaan OCR pada Setiap Nota

Kode program 4.8 adalah kode program yang digunakan untuk melakukan pembacaan OCR pada seluruh nota yang telah disegmentasi. Kode program ini menghasilkan sebuah *file* dengan format *.json* yang menyimpan hasil deteksi pada setiap nota.

4.1.3 Annotasi Dataset

Annotasi dataset adalah proses yang dilakukan untuk memberikan label pada data yang dibuat. Setiap kata yang ada dalam nota yang telah dideteksi, akan diberikan label secara manual dengan pemberian angka sesuai pada tabel 4.1.

```

def annot_helper(file_name,bboxes,words,batch_num):

    file_name = file_name.split('.json')[0]+' .jpg'
    img = cv2.imread(os.path.join(DIR, 'Nota_Segmented',file_name))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    ex_box = []

    for bbox in bboxes:
        bbox = unnormalize_bbox(bbox, img.shape)
        ex_box.append(bbox)

    ex_word = words[batch_num*10:(batch_num+1)*10]
    ex_box = ex_box[batch_num*10:(batch_num+1)*10]

    for box in ex_box:
        cv2.rectangle(img, (box[0], box[1]), (box[2], box[3]),
(255, 0, 0), 1)

    plt.figure(figsize=(10,10))
    plt.imshow(img)

    plt.figure(figsize=(20,10))

    for i in range(len(ex_word)):
        plt.subplot(2,5,i+1)
        plt.imshow(img[ex_box[i][1]:ex_box[i][3],
ex_box[i][0]:ex_box[i][2]])

        plt.title(ex_word[i])

    plt.show()
file_name = '20221013_192759.json'

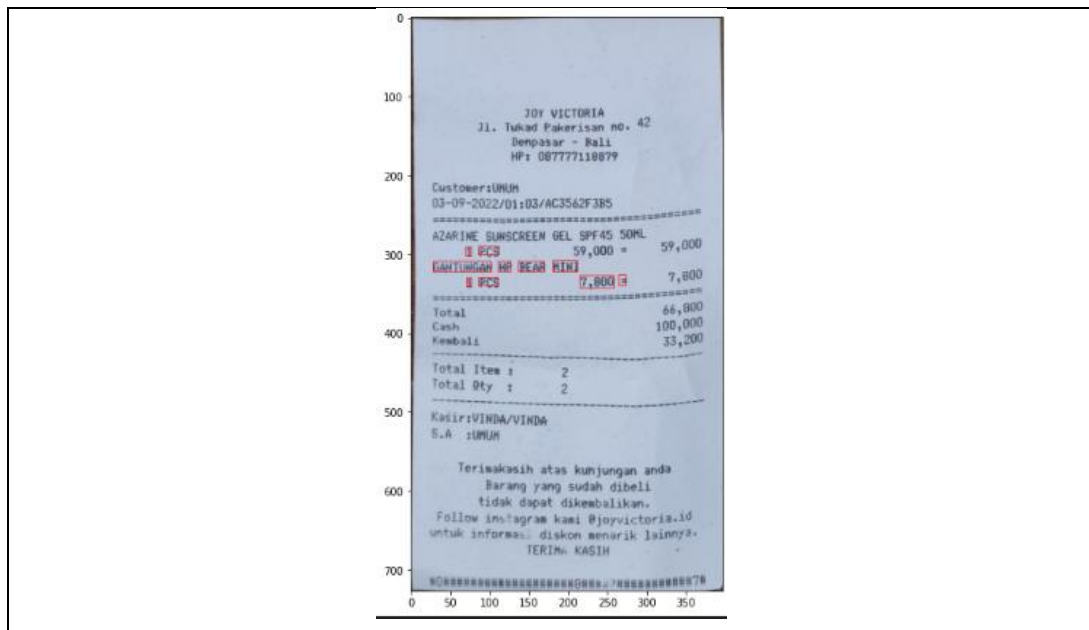
with open(os.path.join(DIR, 'Annotation',file_name)) as f:
    data = json.load(f)

annot_helper(file_name,data['bboxes'],data['words'],batch_num=4)

```

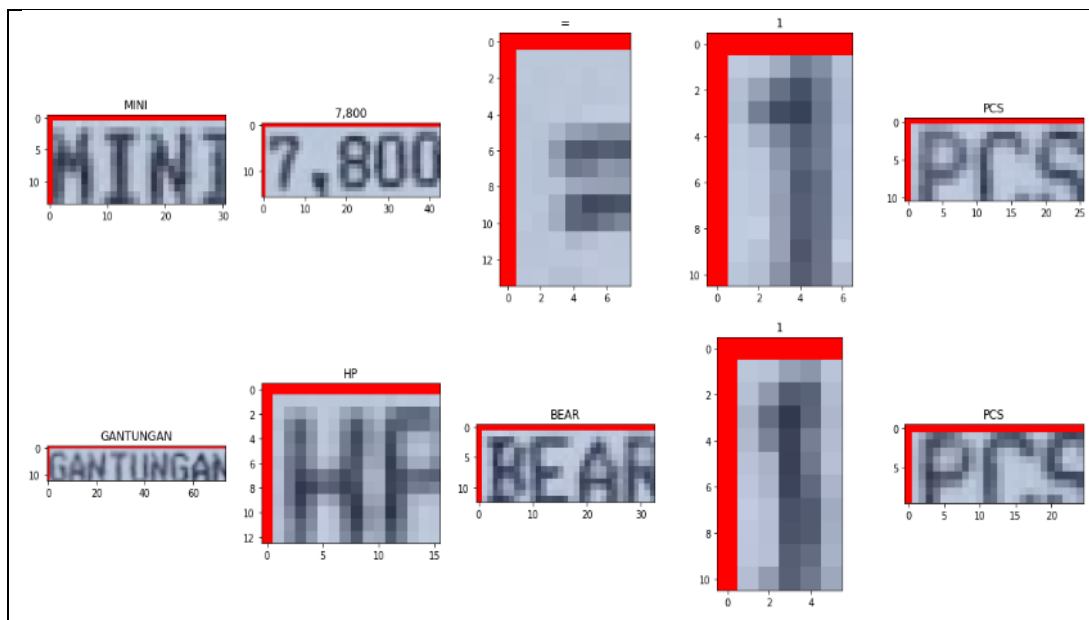
Kode Program 4.9 Fungsi Pembantu Annotasi Dataset

Kode program 4.9 adalah kode program yang digunakan untuk membantu proses *annotasi dataset* pada setiap *file* nota. Program bekerja dengan cara meng-*input*-kan nama *file* pada variabel `file_name` dan memberikan `batch_num` dimulai dari nol. Tujuan penggunaan *batch* adalah agar kata dan letak dari kata tersebut dapat diberikan label yang tepat. Setiap *batch* memiliki 10 kata dan jumlah *batch* pada suatu nota akan menyesuaikan dengan jumlah kata pada nota tersebut.



Gambar 4.8 Gambar Penggunaan Fungsi Annotasi tahap 1

Gambar 4.8 merupakan gambar hasil penggunaan fungsi *annotasi* bantuan. Setiap kotak merah memiliki sebuah kata yang perlu dideteksi dan kata-kata dapat terlihat dengan jelas pada gambar 4.9.



Gambar 4.9 Gambar Penggunaan Fungsi Annotasi tahap 2

Gambar 4.9 adalah gambar penggunaan fungsi *annotasi* bantuan yang berfokus pada setiap kata. Pelabelan dilakukan dengan mengikuti urutan dari kiri atas hingga kanan bawah dengan label yang sesuai.

4.2 Finetuning Model LayoutLM

Proses *Finetuning* dilakukan dengan menggunakan data latih sebanyak 80% hasil pembuatan data yang ditambahkan dengan 1267 data latih dari *dataset* sekunder. Proses pengujian dilakukan dengan menggunakan 20% dari data yang dibuat serta 472 *dataset* sekunder.

4.2.1 Data Preparation

Data preparation adalah tahap persiapan data di mana data dimuat ke dalam program. Setelah data dimuat ke dalam program, data kemudian diproses untuk mendapatkan data latih dan data uji.

```
dataset = load_dataset("Theivaprakasham/wildreceipt")
example = dataset["train"][0]
example["image_path"]

words, bboxes, ner_tags = example["words"], example["bboxes"],
example["ner_tags"]

print(words)
print(bboxes)
print(ner_tags)
```

Kode Program 4.10 *Persiapan Dataset*

Kode program 4.10 adalah kode program yang digunakan untuk memuat data sekunder *WildReceipt*. *Dataset* dimuat dengan menggunakan fungsi `load_dataset` dan diambil kata-kata, *bounding box*, serta label dari *dataset* tersebut.

```

def normalize_bbox(bbox, size):
    return [
        int(bbox[0] * 1000 / size[1]),
        int(bbox[1] * 1000 / size[0]),
        int(bbox[2] * 1000 / size[1]),
        int(bbox[3] * 1000 / size[0]),
    ]

def unnormalize_bbox(bbox, size):
    return [
        int(bbox[0] * size[1] / 1000),
        int(bbox[1] * size[0] / 1000),
        int(bbox[2] * size[1] / 1000),
        int(bbox[3] * size[0] / 1000),
    ]

```

Kode Program 4.11 Fungsi Normalisasi Dataset

Kode program 4.11 adalah kode yang digunakan untuk melakukan normalisasi pada *bounding box* kata dalam *dataset*. Normalisasi dilakukan agar data dapat diproses menggunakan *AutoProcessor* milik Model LayoutLM.

```

img_read = cv2.imread(example["image_path"])
img_read = cv2.cvtColor(img_read, cv2.COLOR_BGR2RGB)

if img_read is None:
    raise Exception(f"Image {example['image_path']} not found")

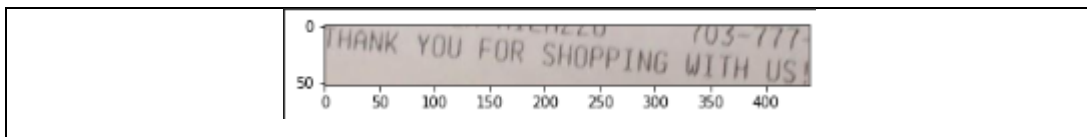
xx = [360, 191, 635, 235]
xx = unnormalize_bbox(xx, img_read.shape)

plt.imshow(img_read[xx[1]:xx[3],xx[0]:xx[2]])
plt.show()
print(img_read.shape)
xx

```

Kode Program 4.12 Segmentasi Kata Bounding Box

Kode program 4.12 adalah kode program yang digunakan untuk melakukan segmentasi kata pada *bounding box*. Kode ini menampilkan contoh kata yang terdapat pada nota dengan segmentasi setelah diterapkan normalisasi *bounding box*.



Gambar 4.10 Hasil Segmentasi Bounding Box

Gambar 4.10 adalah gambar contoh hasil segmentasi *bounding box*. Segmentasi yang tepat pada gambar menunjukkan normalisasi telah berhasil diterapkan pada nota dan siap untuk diproses menggunakan *AutoProcessor* milik LayoutLM.

4.2.2 Data Pipelining

Data pipelining merupakan proses yang digunakan untuk memproses data lebih lanjut untuk dapat dilatih pada Model. Proses ini memanfaatkan *AutoProcessor* milik LayoutLM untuk menyesuaikan format *dataset* serta mengubahnya menjadi *encoding* untuk dilatih pada Model.

```
processor = AutoProcessor.from_pretrained("microsoft/layoutlmv3-
base", apply_ocr=False)
```

Kode Program 4.13 Inisiasi *AutoProcessor* LayoutLM

Kode program 4.13 adalah kode program yang digunakan untuk mendefinisikan *AutoProcessor* milik LayoutLM. *AutoProcessor* ini adalah sebuah *transformer* yang memiliki *input* dan *output* yang sesuai dengan format pelatihan LayoutLM.

```
features = dataset["train"].features
column_names = dataset["train"].column_names
image_column_name = "image_path"

# In the event the labels are not a `Sequence[ClassLabel]`, we
will need to go through the dataset to get the

# unique labels.

def get_label_list(labels):
    unique_labels = set()
    for label in labels:
        unique_labels = unique_labels | set(label)
    label_list = list(unique_labels)
    label_list.sort()
    return label_list
```

```

if isinstance(features["ner_tags"].feature, ClassLabel):
    label_list = features["ner_tags"].feature.names
    # No need to convert the labels since they are already ints.
    id2label = {k: v for k,v in enumerate(label_list)}
    label2id = {v: k for k,v in enumerate(label_list)}

else:

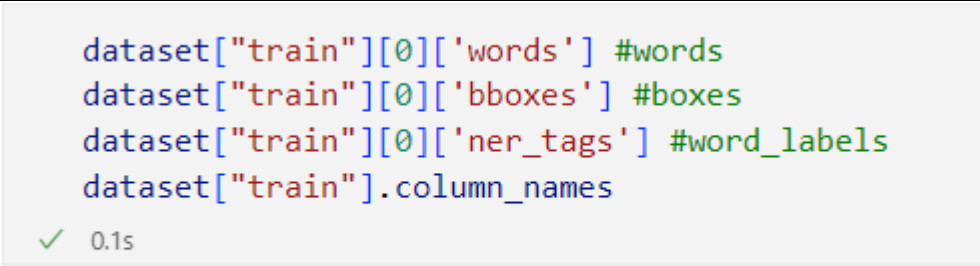
    label_list = get_label_list(dataset["train"]["ner_tags"])
    id2label = {k: v for k,v in enumerate(label_list)}
    label2id = {v: k for k,v in enumerate(label_list)}

num_labels = len(label_list)

```

Kode Program 4.14 *Pengolahan Dataset*

Kode Program 4.14 adalah kode program yang digunakan untuk mengambil informasi yang terdapat pada *dataset*. Kode ini menghasilkan sebuah *list* dengan kumpulan kata, *bounding box*, label, serta letak *file* gambar.



```

dataset["train"][0]['words'] #words
dataset["train"][0]['bboxes'] #boxes
dataset["train"][0]['ner_tags'] #word_labels
dataset["train"].column_names

```

✓ 0.1s

```

['id', 'words', 'bboxes', 'ner_tags', 'image_path']

```

Gambar 4.11 *Hasil Pengolahan Dataset*

Gambar 4.11 adalah gambar hasil pengolahan *dataset*. Pada gambar terlihat bahwa variabel *dataset*, telah memuat informasi-informasi yang dibutuhkan untuk melakukan pelatihan pada Model LayoutLM.


```

def prepare_examples(examples):
    images = [Image.open(path).convert("RGB") for path in
examples['image_path']] #Image.open(examples[image_column_name])

    words = examples["words"]
    boxes = examples["bboxes"]
    word_labels = examples["ner_tags"]

    encoding = processor(images, words, boxes=boxes,
word_labels=word_labels, truncation=True, padding="max_length")

    return encoding

features = Features({
    'pixel_values': Array3D(dtype="float32", shape=(3, 224, 224)),
    'input_ids': Sequence(feature=Value(dtype='int64')),
    'attention_mask': Sequence(Value(dtype='int64')),
    'bbox': Array2D(dtype="int64", shape=(512, 4)),
    'labels': Sequence(ClassLabel(names=label_list)),
})

train_dataset = dataset["train"].map(
    prepare_examples,
    batched=True,
    remove_columns=column_names,
    features=features,
)

eval_dataset = dataset["test"].map(
    prepare_examples,
    batched=True,
    remove_columns=column_names,
    features=features,
)

train_dataset.set_format("torch")

```

Kode Program 4.15 *Pengubahan Data Menjadi Encoding*

Kode program 4.15 adalah kode program yang digunakan untuk mengubah data yang telah dipisahkan menjadi bentuk *encoding* untuk dilanjutkan ke tahap pelatihan Model. Hasil dari *encoding* yang dilakukan adalah dalam format *torch* dengan jumlah data latih sebanyak 1267 dan data uji berjumlah 472.

```

example = train_dataset[0]
for k,v in example.items():
    print(k,v.shape)

```

✓ 0.8s

```

pixel_values torch.Size([3, 224, 224])
input_ids torch.Size([512])
attention_mask torch.Size([512])
bbox torch.Size([512, 4])
labels torch.Size([512])

```

```

train_dataset

```

✓ 0.5s

```

Dataset({
  features: ['pixel_values', 'input_ids', 'attention_mask', 'bbox', 'labels'],
  num_rows: 1267
})

```

Gambar 4.12 Hasil Pengubahan Data Menjadi Encoding

Gambar 4.12 adalah gambar hasil pengubahan data menjadi *encoding*. Gambar ini memperlihatkan contoh data setelah melalui proses *AutoProcessing* milik LayoutLM. Hasil dari *dataset* memiliki fitur berupa *pixel_values*, *input_ids*, *attention_mask*, *bbox*, dan *labels*.

4.2.3 Model Finetuning

Model *Finetuning* adalah tahap pelatihan Model dengan menggunakan data yang telah dipersiapkan. Model ini dilatih dengan menggunakan data latih dan disimpan untuk membaca nota yang ada pada data uji.

```

metric = load_metric("segeval")

return_entity_level_metrics = False

def compute_metrics(p):
    predictions, labels = p
    predictions = np.argmax(predictions, axis=2)

    # Remove ignored index (special tokens)
    true_predictions = [
        [label_list[p] for (p, l) in zip(prediction, label) if l
        != -100]
        for prediction, label in zip(predictions, labels)
    ]

```

```

    true_labels = [
        [label_list[l] for (p, l) in zip(prediction, label) if l
!= -100]
        for prediction, label in zip(predictions, labels)
    ]

    results = metric.compute(predictions=true_predictions,
references=true_labels)

    if return_entity_level_metrics:
        # Unpack nested dictionaries
        final_results = {}

        for key, value in results.items():
            if isinstance(value, dict):
                for n, v in value.items():
                    final_results[f"{key}_{n}"] = v
            else:
                final_results[key] = value

        return final_results
    else:
        return {
            "precision": results["overall_precision"],
            "recall": results["overall_recall"],
            "f1": results["overall_f1"],
            "accuracy": results["overall_accuracy"],
        }

```

Kode Program 4.16 *Metrics Pelatihan Model*

Kode program 4.16 adalah kode program yang digunakan untuk melakukan evaluasi dari pelatihan Model. Evaluasi Model selama pelatihan dilakukan dengan menggunakan fungsi `load_metric` dari *library Datasets* dengan penilaian presisi, *recall*, skor f1, dan akurasi.

```
Model=LayoutLMv3ForTokenClassification.from_pretrained("microsoft/
layoutlmv3-base",id2label=id2label,label2id=label2id)
```

Kode Program 4.17 Memuat Model LayoutLM

Kode Program 4.17 adalah kode program yang digunakan untuk memuat Model LayoutLM dasar. Kode ini memuat Model LayoutLM dasar milik Microsoft dengan parameter sebuah variabel *dictionary* yang berisikan daftar nama label beserta nomor dari nama label tersebut.

```
training_args = TrainingArguments(output_dir="layoutlmv3-
finetuned-wildreceipt",
max_steps=4000,
per_device_train_batch_size=4,
per_device_eval_batch_size=4,
learning_rate=1e-5,
evaluation_strategy="steps",
eval_steps=100,
load_best_Model_at_end=True,
metric_for_best_Model="f1",
)
trainer = Trainer(
    Model=Model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=eval_dataset,
    tokenizer=processor,
    data_collator=default_data_collator,
    compute_metrics=compute_metrics,
)
trainer.train()
```

Kode Program 4.18 Pelatihan Model

Kode program 4.18 merupakan kode program yang digunakan untuk melatih Model. Model dilatih dengan melihat skor f1 tertinggi serta Model terbaik yang ditemukan dimuat secara otomatis pada akhir sesi pelatihan.

4.2.4 Model Evaluation

Model *evaluation* berisi tentang hasil evaluasi pada Model yang telah dilatih sebelumnya. Metode yang digunakan dalam penilaian Model menggunakan *confussion matrix*, serta *classification report* sesuai dengan yang telah dijelaskan pada bab III.

```

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
import seaborn as sns

print(classification_report(y_aktual, y_prediksi, digits=4))

cm = confusion_matrix(y_aktual, y_prediksi)
cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
plt.figure(figsize=(12,12))
plt.title("Train Model 10epoch all data")
sns.heatmap(cm, annot=True, fmt='.2f', xticklabels=[id2label[it]
for it in important_labels], yticklabels=[id2label[it] for it in
important_labels])

```

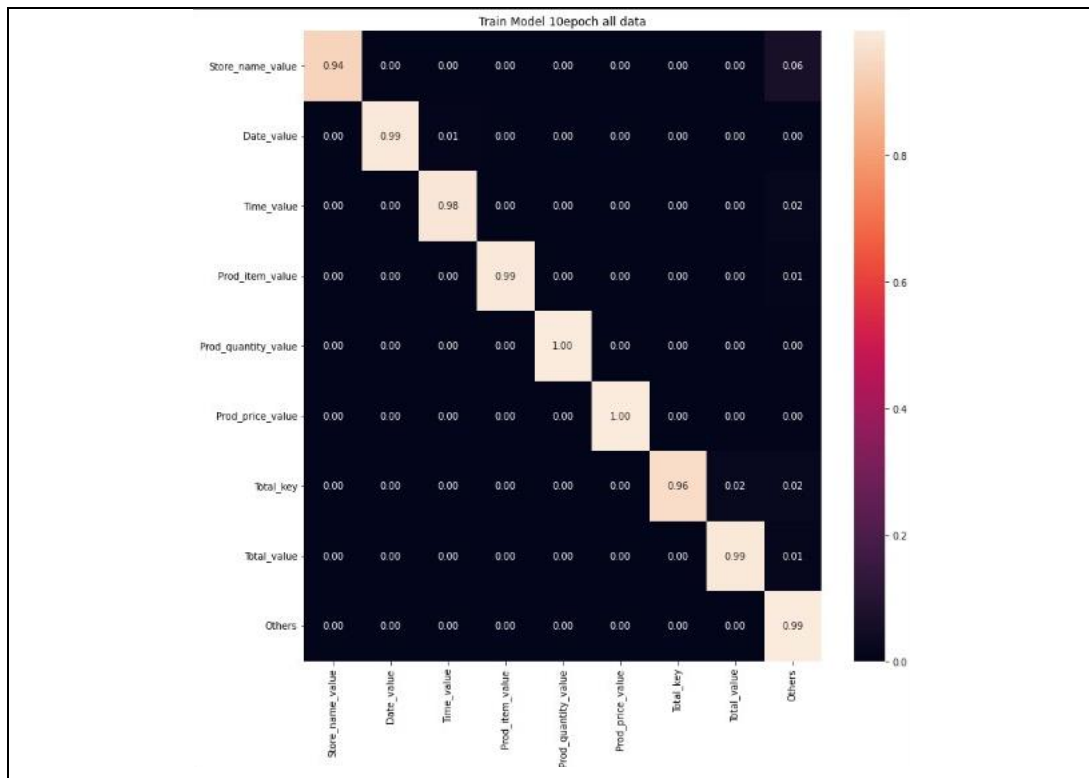
Kode Program 4.19 *Evaluasi Model*

Kode program 4.19 adalah kode program yang digunakan untuk melakukan evaluasi pada hasil pelatihan. Evaluasi dari Model dilihat dengan menggunakan data *training* serta data *testing* untuk melihat perbedaan akurasi dalam pelatihan serta pada data uji.

	precision	recall	f1-score	support
1	0.9851	0.9362	0.9600	141
2	0.9400	0.9895	0.9641	95
3	0.9462	0.9840	0.9647	125
5	0.9909	0.9889	0.9899	993
7	0.9788	0.9957	0.9872	232
9	1.0000	0.9960	0.9980	251
12	0.9794	0.9596	0.9694	99
13	0.9506	0.9872	0.9686	78
14	0.9955	0.9946	0.9951	5775
accuracy			0.9922	7789
macro avg	0.9741	0.9813	0.9774	7789
weighted avg	0.9922	0.9922	0.9922	7789

Gambar 4.13 Hasil Data Latih *Classification Report*

Gambar 4.13 merupakan hasil *classification report* yang didapatkan oleh Model pada data pelatihan yaitu 80 nota berbahasa Indonesia. Model yang dilatih mendapatkan akurasi 99,98 persen yang berarti model berhasil membaca nota dengan sangat baik.



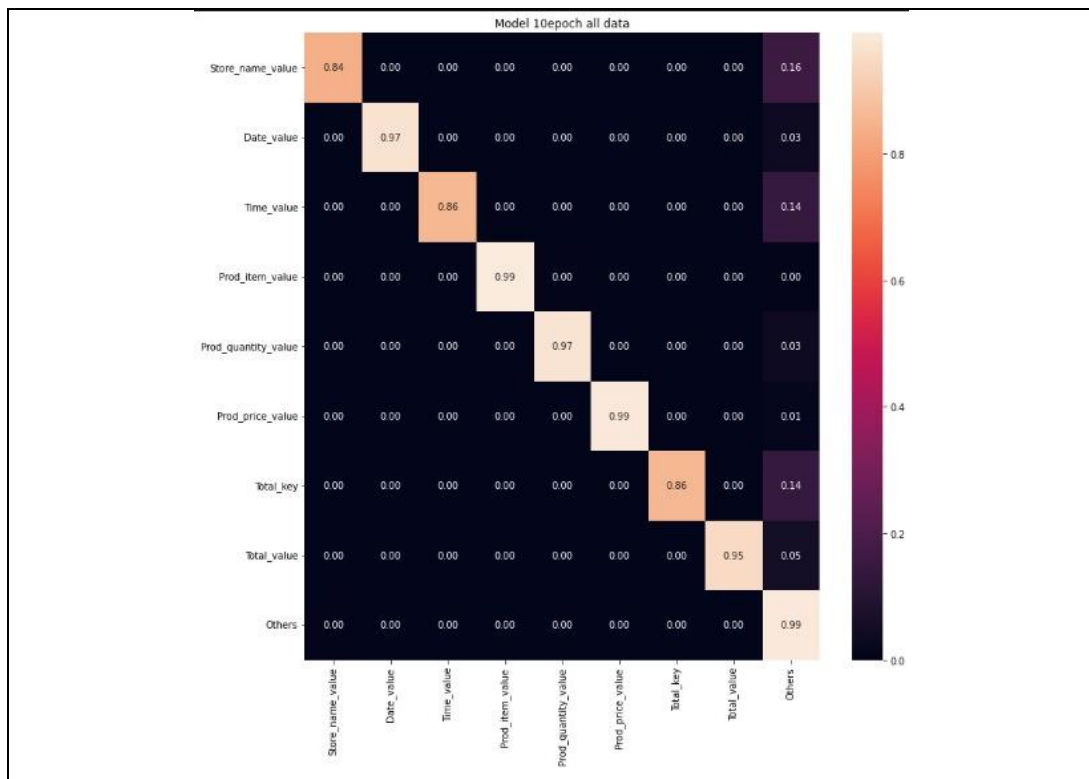
Gambar 4.14 Hasil Data Latih Confussion Matrix

Gambar 4.14 merupakan hasil *confussion matrix* yang didapatkan oleh Model pada data pelatihan. Gambar ini memperlihatkan bahwa model memiliki akurasi yang sangat tinggi pada setiap label.

	precision	recall	f1-score	support
1	0.8378	0.8378	0.8378	37
2	1.0000	0.9655	0.9825	29
3	1.0000	0.8571	0.9231	28
5	0.9790	0.9915	0.9852	235
7	0.9683	0.9683	0.9683	63
9	0.9444	0.9855	0.9645	69
12	0.9600	0.8571	0.9057	28
13	0.9474	0.9474	0.9474	19
14	0.9860	0.9874	0.9867	1425
accuracy			0.9798	1933
macro avg	0.9581	0.9331	0.9446	1933
weighted avg	0.9799	0.9798	0.9797	1933

Gambar 4.15 Hasil data uji *Classification Report*

Gambar 4.13 merupakan hasil *classification report* yang didapatkan oleh Model pada data uji yaitu 20 nota berbahasa Indonesia. Model yang dilatih mendapatkan akurasi sebesar 97,98 persen.



Gambar 4.16 Hasil Data Uji Confussion Matrix

Gambar 4.14 merupakan hasil *confussion matrix* yang didapatkan oleh Model pada data uji. Gamabr ini memperlihatkan bahwa model memiliki akurasi yang tinggi, namun masih memiliki kesalahan dalam membaca label nama toko, total key, serta waktu yang dibaca sebagai label *other*.

4.3 Deployment Sistem

Deployment Sistem dilakukan dengan menerapkan Model yang telah dibuat ke dalam sistem agar Model dapat digunakan untuk membaca nota secara *real-time*. Sistem dibuat dengan *web server* berbasis *framework Flask* serta sebuah aplikasi Android untuk mengirimkan gambar nota pada Model.

4.3.1 Flask API Server

Flask API server adalah *framework* yang digunakan untuk *men-deploy* Model dalam bentuk sebuah API. *Flask server* akan menerima gambar nota yang telah disegmentasi dari *smartphone* Android dan melakukan *inferensi* Model pada gambar nota. Hasil dari proses inferensi kemudian dikirimkan kembali kepada perangkat Android.

```
@app.route('/detect', methods=['POST'])
def detect():
    try:
        data = json.loads(request.data)
        bytearrayimg = decodeB64(data)
        bytearrayimg = bytearray(bytearrayimg)
        pil_image = Image.open(io.BytesIO(bytearrayimg))
        cv_img = np.array(pil_image)

        # Convert RGB to BGR
        cv_img = cv_img[:, :, ::-1].copy()

        path = os.path.join(DIR, 'android_img.jpg')
        cv2.imwrite(path, cv_img)

        inf_img, img_info = process_image(model, processor, filepath
= path)
        img_info = reformatInfo(img_info)
        img_byte_arr = io.BytesIO()
        inf_img.save(img_byte_arr, format='jpeg')
        img_byte_arr = img_byte_arr.getvalue()
        img_str = base64.b64encode(img_byte_arr).decode('utf-8')

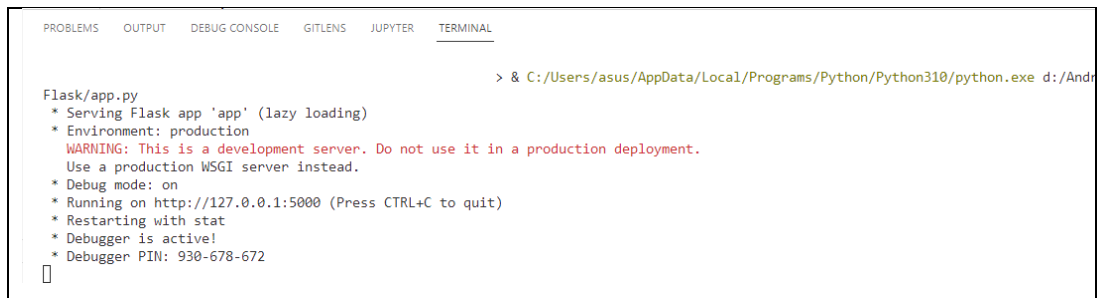
        return
    except Exception as e:
        return badRequest(e, "error")

    successResponse(singleReceipt(img_info), img_str, "success")

if __name__ == '__main__':
    app.run(debug = True)
```

Kode Program 4.20 Web Server Deployment

Kode program 4.20 adalah kode program yang digunakan untuk *men-deploy* Model dalam bentuk *web server* dengan menggunakan *framework Flask*. Gambar terenkripsi yang diterima dari android dimuat dengan menggunakan *library Json* yang kemudian di-*decode* dengan menggunakan *Base64* hingga menjadi format gambar milik *Opencv*. Gambar yang telah terdeskripsi kemudian diproses dengan menggunakan fungsi `process_image`.



```

PROBLEMS OUTPUT DEBUG CONSOLE GIT LENS JUPYTER TERMINAL
> & C:/Users/asus/AppData/Local/Programs/Python/Python310/python.exe d:/Andr
Flask/app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 930-678-672

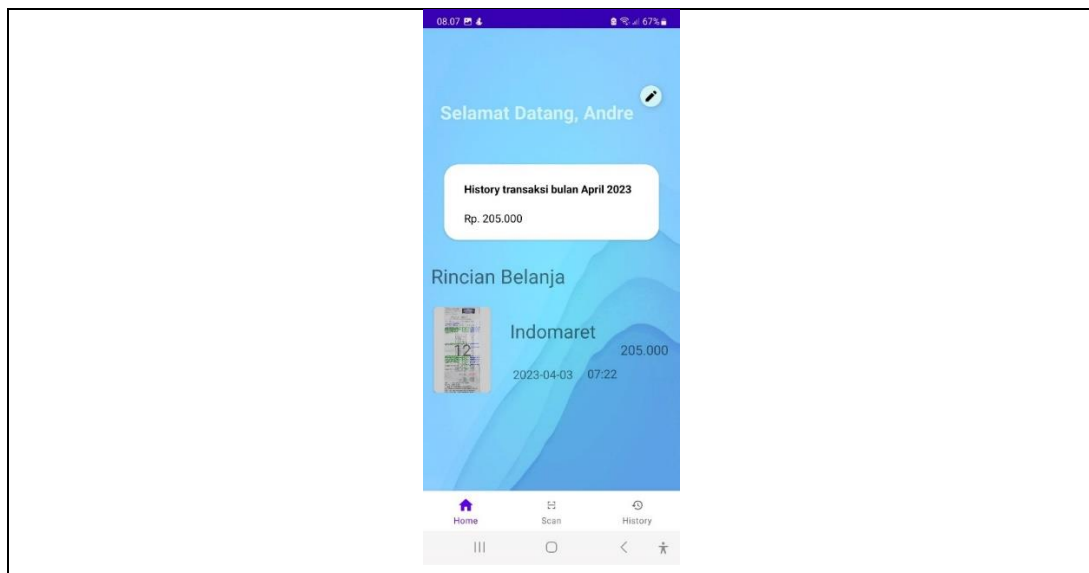
```

Gambar 4.17 Hasil Web Server Deployment

Gambar 4.13 adalah gambar hasil Model *deployment* dengan menggunakan *web server*. Halaman *web server* yang dibuat hanya dapat diakses secara lokal. Aplikasi Android yang dibuat akan memanggil *server* ini untuk menjalankan proses inferensi Model.

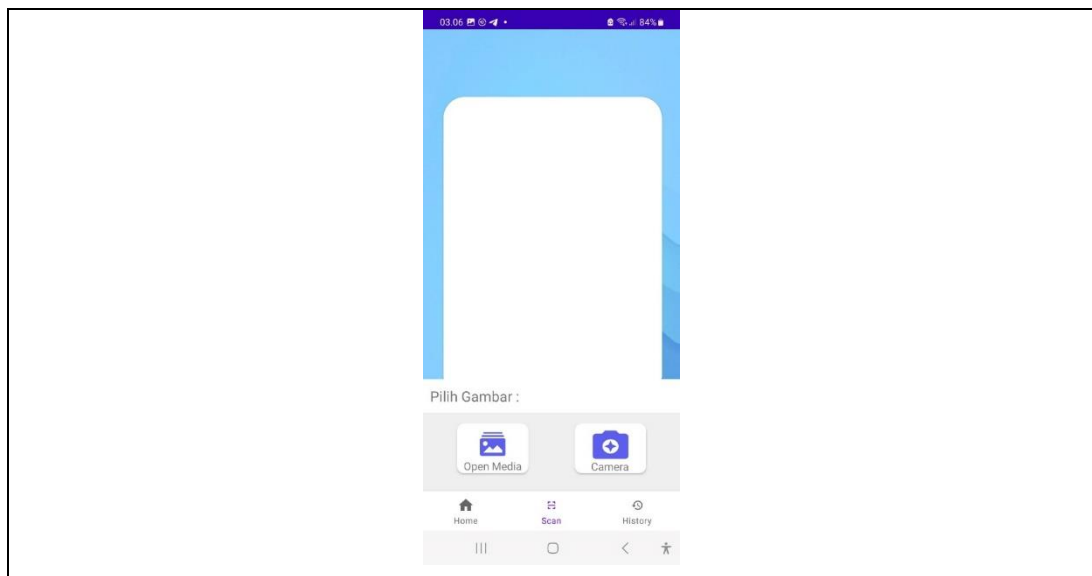
4.3.2 Aplikasi Android

Proses pengujian sistem dilakukan menggunakan sebuah aplikasi Android. Aplikasi yang dibuat memiliki beberapa menu yang mendukung proses inferensi pada nota *server*. Proses yang dilakukan untuk inferensi nota adalah sebagai berikut.



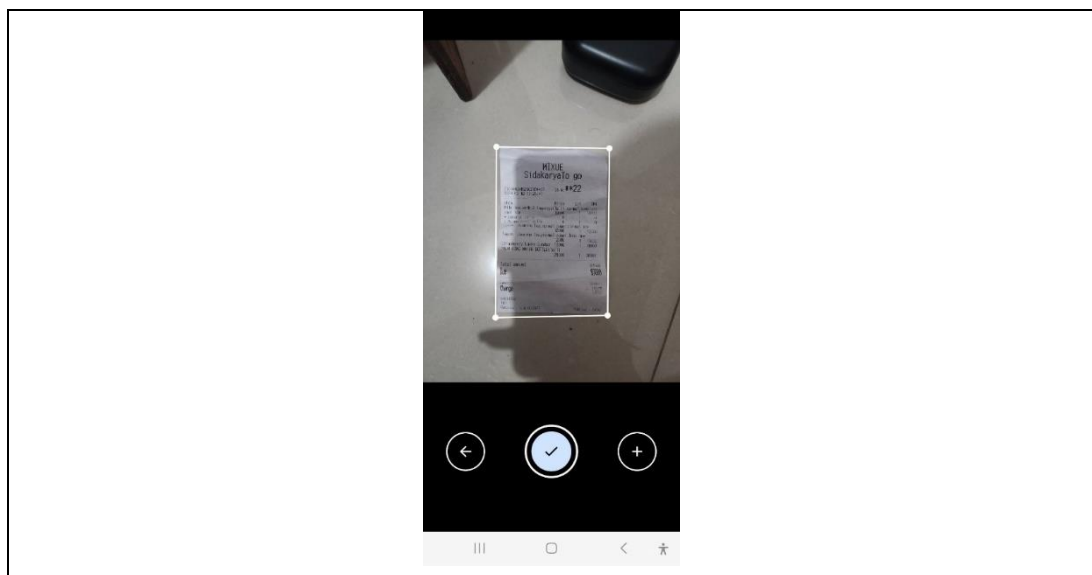
Gambar 4.18 Tampilan Home Aplikasi

Gambar 4.18 adalah gambar tampilan saat pertama kali membuka aplikasi. Tampilan awal dari aplikasi memuat informasi transaksi yang dilakukan selama sebulan, sesuai dengan tanggal aplikasi dibuka.



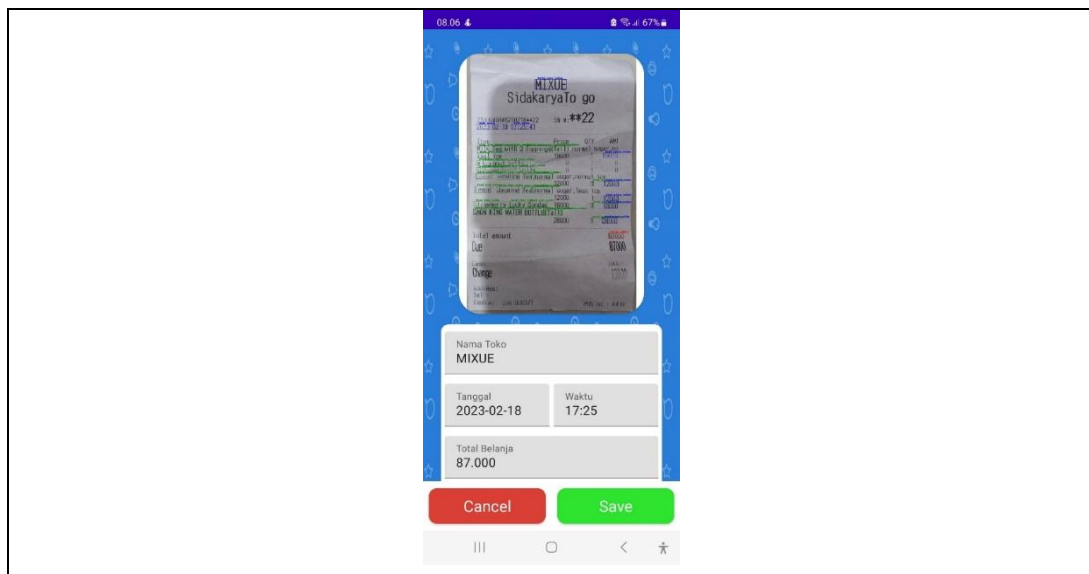
Gambar 4.19 Tampilan Menu Sumber Gambar Nota

Gambar 4.19 adalah gambar tampilan menu *scan* yang memuat pilihan untuk memuat gambar ke dalam sistem. Pilihan menu kamera akan menjalankan kamera yang hasilnya diteruskan pada proses *cropping*.



Gambar 4.20 Tampilan Proses Cropping Pada Aplikasi

Gambar 4.20 merupakan tampilan proses *cropping* yang diterapkan pada gambar. Hasil dari proses pemotongan gambar nota ini selanjutnya dapat diteruskan pada *web server* untuk dibaca.



Gambar 4.21 Tampilan Preview Hasil Pembacaan Nota

Gambar 4.21 merupakan gambar hasil dan informasi yang diterima dari pembacaan nota melalui *web server*. Informasi nama toko, tanggal transaksi, waktu transaksi, total belanja, serta informasi dari setiap produk yang dibeli dimuat dalam tampilan ini. Informasi-informasi ini selanjutnya dapat diperiksa terlebih dahulu sebelum disimpan ke dalam sistem.



4.4 Pengujian Sistem


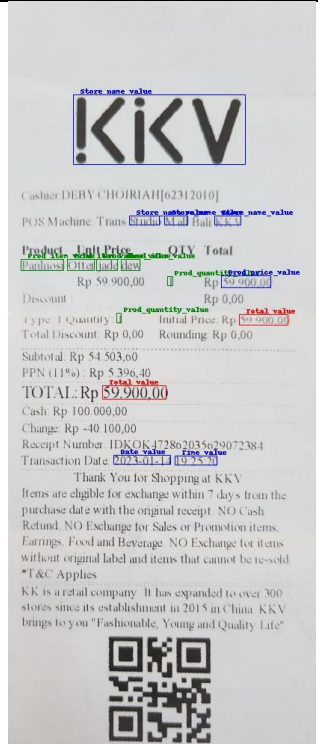
Pengujian sistem membahas tentang uji coba kemampuan sistem dalam mendeteksi dan membaca nota. Skenario pengujian yang digunakan adalah variasi nota secara umum, variasi kecerahan gambar yang dideteksi, serta variasi panjang dari nota yang dibaca. Berikut adalah hasil dari masing-masing skenario pengujian



4.4.1 Pengujian Inferensi Model

Pengujian pertama yaitu inferensi model menguji kemampuan sistem untuk membaca nota yang berasal dari berbagai minimarket dan restoran. Pengujian ini bertujuan untuk mengetahui kemampuan model dalam memahami dan membedakan informasi-informasi penting dalam nota.

Tabel 4.1 Contoh Pengujian Inferensi Model

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat <ul style="list-style-type: none"> • Terdapat salah deteksi 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk dapat terbaca sebagian <ul style="list-style-type: none"> • 1 kuantitas tidak terbaca 5. Total terbaca dengan tepat <p>Akurasi : 85,7%</p>
	<ol style="list-style-type: none"> 1. Nama Toko tidak terdeteksi 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat 5. Total terbaca dengan tepat <p>Akurasi : 93,7%</p>

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian <ul style="list-style-type: none"> • 2 harga tidak terdeteksi • 1 nama produk tidak terdeteksi 5. Total terbaca dengan benar <p>Akurasi : 80%</p>
	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat <ul style="list-style-type: none"> • 1 kuantitas salah terdeteksi 5. Total terbaca dengan tepat <ul style="list-style-type: none"> • 1 total salah terdeteksi <p>Akurasi : 80%</p>

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat 5. Total terbaca dengan tepat <p>Akurasi : 100%</p>
	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat 5. Total terbaca dengan tepat <p>Akurasi : 100%</p>

Tabel 4.1 merupakan tabel contoh pengujian inferensi model. Pengujian ini dilakukan untuk mengukur ketepatan label yang diberikan model pada suatu nota. Terdapat 5 buah item yang perlu diberi label yaitu nama toko, tanggal, waktu, produk, serta total belanja.

Tabel 4.2 Rangkuman Hasil Pengujian Inferensi Model

NO	Nama Toko	Jumlah Uji	Hasil Uji				Akurasi Rate-Rata
1	Indomaret	5	Nama Toko				91,7%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			5/5	0/5	0/5	1	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			5/5	0/5	0/5	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			5/5	0/5	0/5	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			22/25	2/25	1/25	1	
Total Belanja							
Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi				
5/5	0/5	0/5	0				
2	Circle K	3	Nama Toko				93,3%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			0/3	0/3	3/3	0	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			11/11	0/11	0/11	0	
Total Belanja							
Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi				
3/3	0/3	0/3	0				
3	McDonald	3	Nama Toko				86,9%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	

			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			10/13	3/13	0/13	0	
			Total Belanja				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/3	0/3	1/3	0	
4	Twisterdog	3	Nama Toko				90,9%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			1/3	0/3	2/3	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			7/8	1/8	0/8	0	
			Total Belanja				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
5	Alfamart	2	Nama Toko				100%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			7/7	0/7	0/7	0	
			Total Belanja				
			Tepat	Sebagian	Tidak	Salah	

			2/2	0/2	Terdeteksi 0/2	Deteksi 0	
6	Mixue	2	Nama Toko				94,4%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			3/3	0/3	0/3	0	
			Total Belanja				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			2/2	0/2	0/2	1	
7	Lainnya	14	Nama Toko				88,3%
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			14/14	0/14	0/14	4	
			Tanggal				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			12/14	0/14	2/14	0	
			Waktu				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			12/12	0/12	0/12	0	
			Produk				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			31/35	4/35	0/35	7	
			Total Belanja				
			Tepat	Sebagian	Tidak Terdeteksi	Salah Deteksi	
			12/14	0/14	2/14	2	


Tabel 4.2 merupakan rangkuman dari pengujian inferensi model. Masing-masing item memiliki empat tingkatan indikator deteksi yaitu terdeteksi dengan tepat, terdeteksi sebagian, tidak terdeteksi serta salah deteksi. Indikator tepat, terdeteksi sebagian dan tidak terdeteksi mengacu kepada banyaknya item yang terdeteksi model, yang dibandingkan dengan total item yang seharusnya dideteksi model. Salah deteksi mengacu pada kata yang seharusnya diabaikan tetapi terdeteksi sebagai salah satu item pada nota.

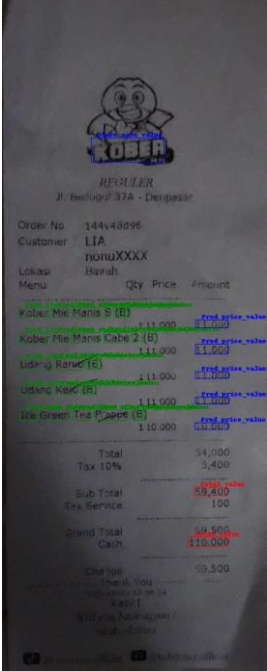
Hasil yang didapatkan dari pengujian ini adalah model dapat membaca informasi yang terdapat pada nota dengan akurasi rata-rata 90%. Model berhasil membaca variasi letak informasi pada nota dengan baik walaupun nota yang dibaca berasal dari berbagai toko yang berbeda. Model masih kesulitan dalam membaca bentuk nota dari beberapa toko, salah satunya adalah nota dari Circle K yang tidak dapat dideteksi nama tokonya.

4.4.2 Pengujian Variasi Kecerahan Nota

Pengujian kedua adalah pengujian kemampuan model dalam membaca gambar nota pada tingkat kecerahan yang berbeda. Variasi pada tingkat kecerahan pada gambar yang diuji berasal dari lingkungan yang gelap, gambar yang tertutupi bayangan tangan, serta lingkungan yang cerah. Pengujian dilakukan sebanyak 5 kali dengan nota yang berbeda. Metode pengujian variasi kecerahan dibagi menjadi dua. Metode pengujian pertama menentukan pengaruh kecerahan gambar nota pada kemampuan deteksi model. Metode pengujian kedua menentukan pengaruh bayangan objek yang terdapat pada gambar nota saat proses pengambilan gambar.

Tabel 4.3 Contoh Pengujian Kecerahan Nota

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat <ul style="list-style-type: none"> • 2 kuantitas terbaca sebagian 5. Total terbaca dengan tepat <p>Akurasi : 89,4%</p>
---	---

	<ol style="list-style-type: none"> 1. Nama Toko tidak dapat terbaca 2. Tanggal tidak dapat terbaca 3. Waktu tidak dapat terbaca 4. Produk terbaca sebagian <ul style="list-style-type: none"> • Kelima kuantitas produk tidak terdeteksi 5. Total tidak dapat terbaca <p>Akurasi : 73,6%</p>
---	---

Tabel 4.3 merupakan tabel contoh pengujian variasi kecerahan pada pembacaan nota. Pengujian ini dilakukan untuk mengukur ketepatan label yang diberikan model pada kondisi kecerahan yang berbeda-beda. Sistem dinilai berhasil bila dapat mentoleransi variasi kecerahan gambar nota pada pemberian label.


Tabel 4.4 Rangkuman Hasil Pengujian Kecerahan Nota



NO	Nama Toko	Label	Terang	Gelap
1	Indomaret	Nama Toko	Tepat	Tidak terbaca
		Tanggal	Tepat	Tepat
		Waktu	Tepat	Tidak terbaca
		Produk	Terbaca Sebagian	Terbaca Sebagian
		Total Belanja	Tepat	Tepat
2	Kober	Nama Toko	Tepat	Tepat
		Tanggal	Tepat	Tepat
		Waktu	Tepat	Tepat
		Produk	Tepat	Terbaca Sebagian
		Total Belanja	Tepat	Tepat
3	Watson	Nama Toko	Tepat	Tepat
		Tanggal	Tidak terbaca	Tidak terbaca
		Waktu	Tepat	Tepat
		Produk	Tepat	Terbaca Sebagian
		Total Belanja	Tidak Terbaca	Tidak Terbaca
4	Alfamart	Nama Toko	Tepat	Tepat
		Tanggal	Tepat	Tepat
		Waktu	Tepat	Tepat
		Produk	Tepat	Tepat

		Total Belanja	Tepat	Tepat
5	Circle K	Nama Toko	Tidak terbaca	Terbaca Sebagian
		Tanggal	Tepat	Tepat
		Waktu	Tepat	Tidak terbaca
		Produk	Tepat	Tepat
		Total Belanja	Tepat	Tidak terbaca

Tabel 4.4 merupakan rangkuman pengujian variasi kecerahan pada pembacaan nota. Hasil yang didapatkan dari pengujian ini adalah kecerahan gambar nota adalah faktor yang penting dalam keberhasilan model. Model berhasil membaca gambar nota yang diambil pada keadaan kurang cahaya pada beberapa variasi nota. Namun, pada bentuk nota dengan informasi yang berdekatan, sistem segmentasi kesulitan untuk membagi kata dengan baik sehingga model gagal dalam memberi label. Selain itu, model tidak dapat membaca gambar nota yang diambil pada keadaan sangat gelap.

Tabel 4.5 Contoh Pengujian Gambar yang Tertutup Bayangan

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat 5. Total terbaca dengan tepat <p>Akurasi : 92,8%</p>
---	---

	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian <ul style="list-style-type: none"> • Ketiga kuantitas tidak terbaca 5. Total terbaca dengan tepat <p>Akurasi : 71,4%</p>
	<ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian <ul style="list-style-type: none"> • satu kuantitas produk tidak terdeteksi 5. Total terbaca dengan tepat <p>Akurasi : 85,7%</p>

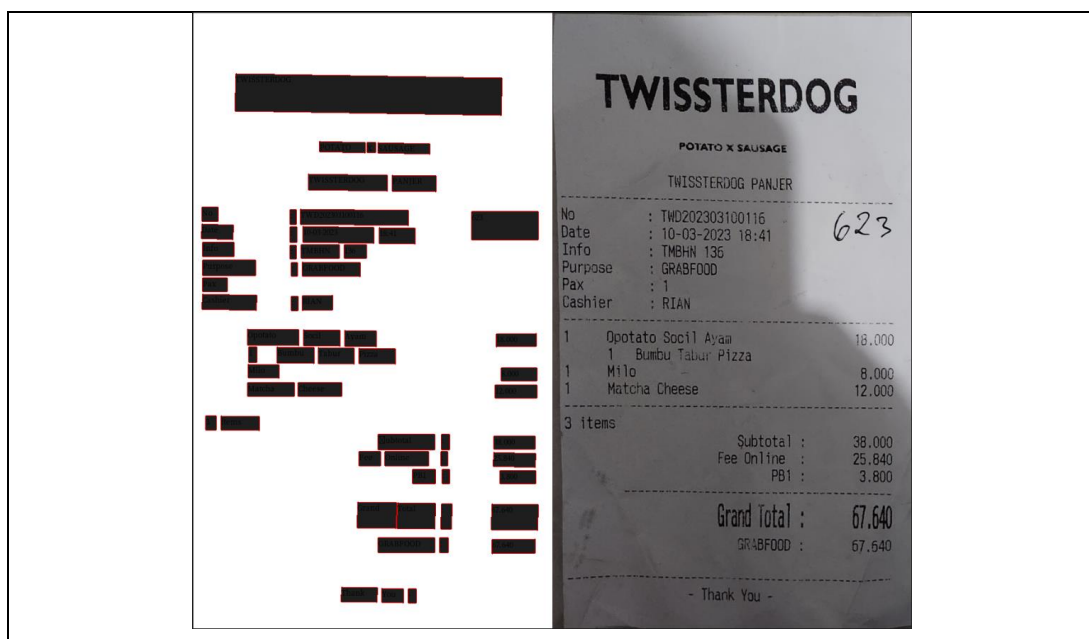
Tabel 4.5 merupakan tabel contoh pengujian inferensi pada gambar yang tertutup bayangan. Pengujian ini dilakukan untuk mengukur ketepatan label yang diberikan model pada kondisi jika objek nota tertutup bayangan saat pengambilan gambar. Sistem dinilai berhasil bila dapat mentoleransi kecerahan yang berbeda pada region nota yang terbaca.

Tabel 4.6 Rangkuman Hasil Pengujian Gambar tertutup bayangan

NO	Nama Toko	Label	Tertutup Bayangan < 30%	Tertutup Bayangan +- 50%	Tertutup Bayangan > 80%
1	Twisterdog	Nama Toko	Tepat	Tepat	Tepat
		Tanggal	Tepat	Tepat	Tepat
		Waktu	Tepat	Tepat	Tepat
		Produk	Tepat	Terbaca Sebagian	Terbaca Sebagian
		Total Belanja	Tepat	Tepat	Tepat
2	Watson	Nama Toko	Tepat	Tepat	Tepat
		Tanggal	Tidak Terbaca	Tidak Terbaca	Tidak Terbaca

		Waktu	Tepat	Tepat	Tepat
		Produk	Tepat	Tepat	Tepat
		Total Belanja	Tidak Terbaca	Tidak Terbaca	Tidak Terbaca
3	Alfamart	Nama Toko	Tepat	Tepat	Tepat
		Tanggal	Tepat	Tepat	Tepat
		Waktu	Tepat	Tepat	Tepat
		Produk	Tepat	Tepat	Terbaca Sebagian
		Total Belanja	Tepat	Tepat	Tepat
4	Circle K	Nama Toko	Terbaca sebagian	Tepat	Tidak Terbaca
		Tanggal	Tepat	Tepat	Tepat
		Waktu	Tepat	Tepat	Tepat
		Produk	Terbaca sebagian	Tepat	Terbaca sebagian
		Total Belanja	Tidak Terbaca	Tidak Terbaca	Tepat

Tabel 4.6 merupakan rangkuman pengujian variasi kecerahan pada gambar nota yang tertutupi bayangan. Berdasarkan hasil analisa yang dilakukan kecerahan pada nota mempengaruhi kemampuan OCR dari *Google Vision* yang membuat beberapa informasi penting seperti kuantitas menjadi tidak dapat terdeteksi.




Gambar 4.22 Contoh Pengaruh Pencahaya pada Deteksi Sistem



Gambar 4.22 menunjukkan nota pada sampel pertama pada tahap pembacaan OCR oleh *Google Vision*. Terlihat pada gambar ketiga kuantitas dari produk yang dibeli tidak terdeteksi sebagai sebuah kata.

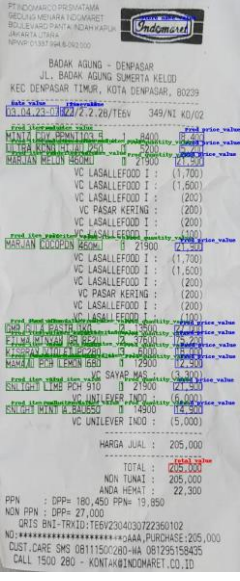
4.4.3 Pengujian Variasi Panjang Nota

Pengujian ketiga adalah pengujian kemampuan model dalam membaca nota dengan ukuran panjang yang berbeda. Pengujian ini dilakukan untuk mengetahui batas panjang nota yang dapat dibaca oleh model. Variasi dalam ukuran panjang nota dapat berasal dari jumlah produk yang dibeli dalam nota serta desain dari nota tersebut.

Tabel 4.7 Tabel Pengujian Variasi Panjang Nota

	<p>Ukuran nota :</p> <ul style="list-style-type: none"> • Panjang : 3722px • Lebar : 504px <p>Jumlah Produk : 31 item</p> <ol style="list-style-type: none"> 1. Nama Toko tidak terbaca 2. Tanggal tidak terbaca 3. Waktu tidak terbaca 4. Produk tidak terbaca <ul style="list-style-type: none"> • 21 produk terbaca sebagian • 10 produk tidak terdeteksi 5. Total tidak terbaca
--	---

	<p>Ukuran nota :</p> <ul style="list-style-type: none"> Panjang : 3110px Lebar : 503px <p>Jumlah Produk : 28 item</p> <ol style="list-style-type: none"> Nama Toko terbaca dengan tepat Tanggal tidak terbaca Waktu tidak terbaca Produk terbaca sebagian <ul style="list-style-type: none"> 20 produk terbaca dengan tepat 8 produk terbaca sebagian Total tidak terbaca
	<p>Ukuran nota :</p> <ul style="list-style-type: none"> Panjang : 2213px Lebar : 679px <p>Jumlah Produk : 20 item</p> <ol style="list-style-type: none"> Nama Toko tidak terbaca Tanggal terbaca dengan tepat Waktu terbaca dengan tepat Produk terbaca sebagian <ul style="list-style-type: none"> 17 produk terbaca sebagian 3 produk tidak terdeteksi Total tidak terbaca

	<p>Ukuran nota :</p> <ul style="list-style-type: none"> • Panjang : 2334px • Lebar : 974px <p>Jumlah Produk : 10 item</p> <ol style="list-style-type: none"> 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat <ul style="list-style-type: none"> • 1 produk terbaca sebagian 5. Total terbaca dengan benar
---	---

Tabel 4.7 merupakan tabel pengujian variasi tinggi nota. Hasil yang didapatkan dari pengujian ini adalah model dapat membaca nota dengan baik pada nota dengan 20 item atau ukuran tinggi gambar asli 2000 *pixel*. Gambar nota yang terlalu besar dan dikompresi akan membuat model kesulitan dalam membaca nota. Selain itu, beberapa variasi nota dengan ukuran *font* yang kecil mempersulit model untuk mengenali informasi nama toko, tanggal, waktu, serta total belanja.

4.4.4 Hasil Pengujian Sistem

Berdasarkan ketiga pengujian yang dilakukan menunjukkan bahwa akurasi model yang didapatkan ketika proses evaluasi awal yaitu 97,98% mengalami fluktuasi dalam pengujian secara *realtime*. Model yang dibuat masih belum dapat bekerja dengan optimal pada beberapa variasi nota yang memiliki letak informasi yang berdekatan, *font* yang terlalu kecil, serta warna nota yang pudar. Selain itu, model juga belum dapat mengekstrak informasi dengan benar pada nota yang terlalu panjang.

Permasalahan lain yang ditemukan selama pengujian sistem adalah sistem belum dapat mendeteksi nota dengan optimal pada kondisi gelap atau nota yang tertutup bayangan. Permasalahan ini disebabkan oleh proses ekstraksi OCR melalui *Google Vision* yang terkadang tidak berhasil mengekstrak informasi penting pada nota.

Sistem deteksi nota dengan menggunakan Model LayoutLM dengan OCR dari *Google Vision* berhasil melakukan ekstraksi informasi pada nota tanpa perlu mengenali *template* dari setiap nota yang ingin dibaca. Penerapan Model LayoutLM dapat menggantikan proses estimasi posisi informasi atau proses *rough estimation* yang dilakukan pada penelitian sistem pengenalan nota otomatis milik Lin (Lin et al., 2022).

4.4.5 Kelebihan dan Kekurangan Sistem

Kelebihan dan kekurangan dari sistem yang telah diimplementasikan, selanjutnya dirangkum pada subbab berikut. Kelebihan dan kekurangan sistem ini didapatkan dari hasil pengujian sistem.

4.4.5.1 Kelebihan Sistem

Kelebihan dari sistem yang ditemukan selama pengujian adalah sebagai berikut.

1. Sistem dapat membaca beragam nota yang berasal dari minimarket dan restoran dengan cukup baik.
2. Sistem dapat mentoleransi nota yang lecek dan tercoret pada saat membaca nota dengan cukup baik.
3. Sistem menyediakan fitur *cropping* serta memilih galeri yang membuat pengguna tidak perlu melakukan *cropping* pada gambar nota dengan bantuan alat lainnya.
4. Sistem dapat menyimpan hasil dari pembacaan nota yang dapat dilihat secara digital pada menu histori.
5. Penyimpanan yang dilakukan pada sistem sudah berupa informasi digital yang berarti total belanja dalam kurun waktu tertentu dapat dilihat dengan menerapkan filter pada sistem.

4.4.5.2 Kekurangan Sistem

Kekurangan dari sistem yang ditemukan selama pengujian adalah sebagai berikut.

1. Informasi yang terdapat pada nota harus utuh (tidak robek, terlipat, tertutupi dsb.) agar nota dapat terbaca dengan baik.
2. Nota yang dibaca oleh sistem tidak boleh miring.
3. Sistem tidak dapat membaca informasi diskon yang terdapat pada nota. Ketidakmampuan sistem ini memungkinkan adanya kesalahan informasi total belanja yang tidak sama dengan jumlah kumulatif dari setiap produk yang dibeli.
4. Kecerahan gambar nota yang diambil dapat mempengaruhi hasil pembacaan sistem.
5. Sistem tidak dapat membaca nota yang panjang dengan baik.
6. Sistem hanya memiliki lima buah variasi letak produk, kuantitas, dan harga produk. Jumlah variasi yang terbatas ini membuat nota baru dengan variasi yang berbeda harus didaftarkan terlebih dahulu pada sistem agar dapat terbaca dengan baik.

BAB V

PENUTUP

5.1 Kesimpulan

Simpulan yang dapat diambil setelah dilaksanakannya penelitian tentang *finetuning* Model LayoutLM untuk membaca nota berbahasa Indonesia ini dapat dijabarkan sebagai berikut.

1. Pembuatan sistem pembaca nota secara otomatis berhasil dilakukan dengan menggunakan Model LayoutLM serta dengan bantuan *Google Vision OCR. Finetuning* yang dilakukan pada Model LayoutLM berhasil membaca nota berbahasa Indonesia dengan akurasi 97,98%.
2. Sistem yang dibuat masih belum dapat mengekstrak informasi dari semua variasi nota dengan benar. Variasi bentuk nota yang beragam membuat Model LayoutLM kesulitan untuk memprediksi label dari setiap kata dengan benar. Selain itu, variasi pada *layout* nota juga mempersulit proses ekstraksi informasi yang telah dikenali oleh model. Variasi dari bentuk nota ini termasuk tapi tidak terbatas pada panjang nota, letak nama toko, letak diskon, penulisan barang, variasi nama indikator total belanja, dll.

5.2 Saran

Saran yang dapat diberikan dari pelaksanaan penelitian ini, baik secara penulisan, teknik, alur penelitian, maupun implementasi dapat dijabarkan sebagai berikut.

1. Penambahan data primer perlu dilakukan agar sistem dapat membaca nota yang panjang serta memahami letak informasi nama toko, tanggal, total belanja, dan daftar produk belanjaan dengan lebih baik.
2. Normalisasi *bounding box* perlu dilakukan dengan mempertimbangkan panjang nota yang ingin dibaca. Normalisasi perlu dilakukan karena adanya limitasi ukuran *bounding box* dari Model LayoutLM yaitu sebesar

1000 x 1000 *pixel*. Alternatif lain untuk mengatasi limitasi ini adalah dengan mengubah ukuran gambar nota. Namun, ukuran gambar yang terlalu kecil akan membuat akurasi proses pembacaan OCR menurun.

DAFTAR PUSTAKA

- Andreas, Y., Gunadi, K., & Purbowo, A. N. (2020). Implementasi Tesseract OCR untuk Pembuatan Aplikasi Pengenalan Nota pada Android. *Jurnal Infra*, 8(1), 2–7.
- Baker, P., & Collins, L. (2023). Creating and analysing a multimodal corpus of news texts with Google Cloud Vision’s automatic image tagger. *Applied Corpus Linguistics*, 3(1), 100043. <https://doi.org/10.1016/j.acorp.2023.100043>
- Darma, I Wayan Agus Surya. (2019). Implementation of Zoning and K-Nearest Neighbor in Character Recognition of Wrésastra Script. *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, 9. <https://doi.org/10.24843/lkjiti.2019.v10.i01.p02>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. <http://arxiv.org/abs/1810.04805>
- Hajiali, M., Fonseca Cacho, J. R., & Taghva, K. (2022). Generating Correction Candidates for OCR Errors using BERT Language Model and FastText SubWord Embeddings. *Lecture Notes in Networks and Systems*, 283, 1045–1053. https://doi.org/10.1007/978-3-030-80119-9_69
- Holeček, M. (2020). *Learning from similarity and information extraction from structured documents*. <https://doi.org/10.1007/s10032-021-00375-3>
- Indrawan, Gede, Asroni, Ahmad, Joni Erawati Dewi, Luh, Gunadi, I Gede Aris, & Paramarta, I Ketut (2022). Balinese Script Recognition Using Tesseract Mobile Framework. *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, 13(3), 160. <https://doi.org/10.24843/LKJITI.2022.v13.i03.p03>
- Kumar, V., Kaware, P., & Singh, P. (2020). *Extraction of information from bill receipts using optical character recognition*.

- Kumar, V., Kaware, P., Singh, P., Sonkusare, R., & Kumar, S. (2020). Extraction of information from bill receipts using optical character recognition. *Proceedings - International Conference on Smart Electronics and Communication, ICOSEC 2020, Icosec*, 72–77. <https://doi.org/10.1109/ICOSEC49089.2020.9215246>
- Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4), 1234–1240. <https://doi.org/10.1093/bioinformatics/btz682>
- Lin, C. J., Liu, Y. C., & Lee, C. L. (2022). Automatic Receipt Recognition System Based on Artificial Intelligence Technology. *Applied Sciences (Switzerland)*, 12(2). <https://doi.org/10.3390/app12020853>
- Liu, Y., James, H., Gupta, O., & Raviv, D. (2022). MRZ code extraction from visa and passport documents using convolutional neural networks. *International Journal on Document Analysis and Recognition (IJDAR)*, 25(1), 29–39. <https://doi.org/10.1007/s10032-021-00384-2>
- Markewich, L., Zhang, H., Xing, Y., Lambert-Shirzad, N., Jiang, Z., Lee, R. K.-W., Li, Z., & Ko, S.-B. (2022). Segmentation for document layout analysis: not dead yet. *International Journal on Document Analysis and Recognition (IJDAR)*, 25(2), 67–77. <https://doi.org/10.1007/s10032-021-00391-3>
- Memon, J., Sami, M., & Khan, R. A. (2019). *Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)*. <http://arxiv.org/abs/2001.00139>
- Ngurah, Gusti, Riantama, Sanditya, Piarsa, Nyoman, Made, Gusti, & Sasmita, Arya (2019). Pengaruh Segmentasi Terhadap Hasil Rotasi Citra Menggunakan Metode Minimum Area Rectangle. *MERPATI*, 7(2).
- Nguyen, D.-D. (2022). TableSegNet: a fully convolutional network for table detection and segmentation in document images. *International Journal on Document Analysis and Recognition (IJDAR)*, 25(1), 1–14. <https://doi.org/10.1007/s10032-021-00390-4>

- Puspitarani, Y, & Syukriyah, Y. (2017). *Pemanfaatan Optical Character Recognition Dan Text Feature Extraction Untuk Membangun Basisdata Pengaduan Tenaga Kerja*. 1(3), 704–710.
- Qaroush, A., Awad, A., Modallal, M., & Ziq, M. (2022). Segmentation-based, omnifont printed Arabic character recognition without font identification. *Journal of King Saud University - Computer and Information Sciences*, 34(6), 3025–3039. <https://doi.org/10.1016/j.jksuci.2020.10.001>
- Ramsay, B., Ralescu, A., van der Knaap, E., & Visa, S. (2011). *Confusion Matrix-based Feature Selection*. *Confusion Matrix-based Feature Selection*. <https://www.researchgate.net/publication/220833270>
- Rao, Z., Zeng, C., Wu, M., Wang, Z., Zhao, N., Liu, M., & Wan, X. (2018). Research on a handwritten character recognition algorithm based on an extended nonlinear kernel residual network. *KSII Transactions on Internet and Information Systems*, 12(1), 413–435. <https://doi.org/10.3837/tiis.2018.01.020>
- Raoui-Outach, R., Million-Rousseau, C., Benoit, A., & Lambert, P. (2018). Deep learning for automatic sale receipt understanding. *Proceedings of the 7th International Conference on Image Processing Theory, Tools and Applications, IPTA 2017, 2018-Janua*, 1–6. <https://doi.org/10.1109/IPTA.2017.8310088>
- Saputra, Kurniawan Dwi, Rahmaastri, Della Anggi, Setiawan, Karina, Suryani, Dewi, & Purnama, Yudy (2019). Mobile financial management application using google cloud vision API. *Procedia Computer Science*, 157, 596–604. <https://doi.org/10.1016/j.procs.2019.09.019>
- Sastrawan, I Kadek, Bayupati, I Putu Agung, & Arsa, Dewa Made Sri (2022). Detection of fake news using deep learning CNN–RNN based methods. *ICT Express*, 8(3), 396–408. <https://doi.org/10.1016/j.ict.2021.10.003>
- Shen, Z., Zhang, R., Dell, M., Lee, B. C. G., Carlson, J., & Li, W. (2021). *LayoutParser: A Unified Toolkit for Deep Learning Based Document Image Analysis*. <http://arxiv.org/abs/2103.15348>

- Smith, M. B., Sparks, H., Almagro, J., Chaigne, A., Behrens, A., Dunsby, C., & Salbreux, G. (2023). Active mesh and neural network pipeline for cell aggregate segmentation. *Biophysical Journal*. <https://doi.org/10.1016/j.bpj.2023.03.038>
- Sudana, Oka, Gunaya, I Wayan, & Putra, I Ketut Gede Darma. (2020). Handwriting identification using deep convolutional neural network method. *Telkomnika (Telecommunication Computing Electronics and Control)*, 18(4), 1934–1941. <https://doi.org/10.12928/TELKOMNIKA.V18I4.14864>
- Ting, Kai Ming (2010). Confusion Matrix. In G. I. Sammut Claude and Webb (Ed.), *Encyclopedia of Machine Learning* (p. 209). Springer US. https://doi.org/10.1007/978-0-387-30164-8_157
- Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., & Zhou, M. (2020). LayoutLM: Pre-training of Text and Layout for Document Image Understanding. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1192–1200. <https://doi.org/10.1145/3394486.3403172>
- Google Cloud. (2022). *Vision AI*. Diambil kembali dari Cloud Vision API: dikutip tanggal 11 Juni 2022, <<https://cloud.google.com/vision>>.
- Theivaprakasham. (2022, Juni 11). *Theivaprakasham/wildreceipt*. Diambil kembali dari Hugging Face: dikutip tanggal 11 Juni 2022, <<https://huggingface.co/datasets/Theivaprakasham/wildreceipt>>.

HALAMAN BELAKANG LAINNYA