**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, Mei 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. Ibu Prof. Ir. Linawati, M.Eng., Sc. 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. Kompiang 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, Mei 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, Natural Language Processing, LayoutLM, Struk*

# ABSTRACT

Shopping is a transaction that generates a record as a payment receipt. Typically, a receipt is given as a small piece of paper that can be easily lost. It is essential to store the transaction information in the receipt digitally. Keeping the information in a digital form will make it easily accessible and will overcome the problem of easily lost receipts. Currently, the process of transferring receipt information into digital form is still being done manually. Having a system that can extract this information helps speed up the digitalization process tremendously. This research proposes a method that applies finetuning to the LayoutLM Model and, with the help of OCR from Google Vision, can be used to extract transaction information contained in the receipt. The finetuned LayoutLM Model successfully achieved an accuracy of 97.98% on training data and 90% accuracy on real-time test scenarios for extracting information on receipts written in the Indonesian language. However, variations in the diverse forms of receipts make extracting information on interrelated labels such as product names, quantities, and prices difficult.

**Kata Kunci:** *Computer Vision, Optical Character Recognition, Natural Language Processing, LayoutLM, Struk*

# DAFTAR ISI

[PERNYATAAN iii](#_Toc135302815)

[KATA PENGANTAR iv](#_Toc135302816)

[ABSTRAK v](#_Toc135302817)

[ABSTRACT vi](#_Toc135302818)

[DAFTAR ISI vii](#_Toc135302819)

[DAFTAR GAMBAR x](#_Toc135302820)

[DAFTAR KODE PROGRAM xii](#_Toc135302821)

[DAFTAR TABEL xiii](#_Toc135302822)

[BAB I PENDAHULUAN 1](#_Toc135302823)

[1.1 Latar Belakang 1](#_Toc135302824)

[1.2 Rumusan Masalah 3](#_Toc135302825)

[1.3 Tujuan Penelitian 3](#_Toc135302826)

[1.4 Manfaat Penulisan 3](#_Toc135302827)

[1.5 Batasan Masalah 4](#_Toc135302828)

[1.6 Sistematika Penulisan 4](#_Toc135302829)

[1.6.1 Bab I Pendahuluan 4](#_Toc135302830)

[1.6.2 Bab II Tinjauan Pustaka 4](#_Toc135302831)

[1.6.3 Bab III Metodologi Penelitian 5](#_Toc135302832)

[1.6.4 Bab IV Pembahasan dan Analisis Hasil 5](#_Toc135302833)

[1.6.5 Bab V Penutup 5](#_Toc135302834)

[BAB II TINJAUAN PUSTAKA 6](#_Toc135302835)

[2.1 State of the Art 6](#_Toc135302836)

[2.2 Segmentasi 17](#_Toc135302837)

[2.3 Layout Parser 17](#_Toc135302838)

[2.4 OCR 17](#_Toc135302839)

[2.5 Google Vision 18](#_Toc135302840)

[2.6 LayoutLM 18](#_Toc135302841)

[2.7 Metriks Evaluasi Model 20](#_Toc135302842)

[BAB III METODOLOGI PENELITIAN 23](#_Toc135302843)

[3.1 Tempat dan Waktu Penelitian 23](#_Toc135302844)

[3.2 Data Penelitian 23](#_Toc135302845)

[3.2.1 Data Primer 23](#_Toc135302846)

[3.2.2 Data Sekunder 24](#_Toc135302847)

[3.3 Instrumen Pembuatan Sistem 24](#_Toc135302848)

[3.4 Alur Penelitian 25](#_Toc135302849)

[3.4.1 Pembuatan Dataset 26](#_Toc135302850)

[3.4.2 Finetuning Model LayoutLM 32](#_Toc135302851)

[3.4.3 Percobaan Inferensi Model 33](#_Toc135302852)

[3.4.4 Evaluasi Model 36](#_Toc135302853)

[3.4.5 Skenario Pengujian Sistem 36](#_Toc135302854)

[3.5 Gambaran Umum Sistem 37](#_Toc135302855)

[3.6 Alur Aplikasi 37](#_Toc135302856)

[3.7 Rancangan Sistem 41](#_Toc135302857)

[BAB IV HASIL DAN PEMBAHASAN 45](#_Toc135302858)

[4.1 Pembuatan Dataset 45](#_Toc135302859)

[4.1.1 Segmentasi Gambar 45](#_Toc135302860)

[4.1.2 Pembacaan Karakter 52](#_Toc135302861)

[4.1.3 Annotasi Dataset 53](#_Toc135302862)

[4.2 Finetuning Model LayoutLM 56](#_Toc135302863)

[4.2.1 Data Preparation 56](#_Toc135302864)

[4.2.2 Data Pipelining 57](#_Toc135302865)

[4.2.3 Model Finetuning 60](#_Toc135302866)

[4.2.4 Model Evaluation 62](#_Toc135302867)

[4.3 Deployment Sistem 65](#_Toc135302868)

[4.3.1 Flask API Server 65](#_Toc135302869)

[4.3.2 Aplikasi Android 66](#_Toc135302870)

[4.4 Pengujian Sistem 69](#_Toc135302871)

[4.4.1 Pengujian Inferensi Model 69](#_Toc135302872)

[4.4.2 Pengujian Variasi Kecerahan Nota 74](#_Toc135302873)

[4.4.3 Pengujian Variasi Panjang Nota 79](#_Toc135302874)

[4.4.4 Hasil Pengujian Sistem 81](#_Toc135302875)

[4.4.5 Kelebihan dan Kekurangan Sistem 82](#_Toc135302876)

[4.5 Perbandingan Hasil 83](#_Toc135302877)

[BAB V PENUTUP 85](#_Toc135302878)

[5.1 Kesimpulan 85](#_Toc135302879)

[5.2 Saran 85](#_Toc135302880)

[DAFTAR PUSTAKA 86](#_Toc135302881)

[HALAMAN BELAKANG LAINNYA 86](#_Toc135302882)

# DAFTAR GAMBAR

[Gambar 2.1 Pelatihan LayoutLM (Xu et al., 2020) 19](#_Toc135801899)

[Gambar 2.2 Input Embedding Model LayoutLM 20](#_Toc135801900)

[Gambar 3.1 Flowchart Alur Penelitian 26](#_Toc135801901)

[Gambar 3.2 Flowchart Pembuatan Dataset 27](#_Toc135801902)

[Gambar 3.3 Proses Segmentasi ROI Nota 28](#_Toc135801903)

[Gambar 3.4 Gambar Hasil Proses ROI 29](#_Toc135801904)

[Gambar 3.5 Gambar Hasil Pembacaan Google Vision 30](#_Toc135801905)

[Gambar 3.6 Gambar Hasil Proses Annotasi Dataset 31](#_Toc135801906)

[Gambar 3.7 Flowchart Pelatihan LayoutLM 33](#_Toc135801907)

[Gambar 3.8 Percobaan Inferensi Model 34](#_Toc135801908)

[Gambar 3.9 Hasil Preprocessing Inferensi Nota 34](#_Toc135801909)

[Gambar 3.10 Hasil Inferensi LayoutLM 35](#_Toc135801910)

[Gambar 3.11 Hasil Deteksi Sistem 35](#_Toc135801911)

[Gambar 3.12 Gambaran Umum Sistem 37](#_Toc135801912)

[Gambar 3.13 Flowchart Pembacaan Nota pada Aplikasi Android 38](#_Toc135801913)

[Gambar 3.14 Flowchart Proses Pembacaan Nota pada Web Server 39](#_Toc135801914)

[Gambar 3.15 Flowchart Preprocessing Gambar pada Server 40](#_Toc135801915)

[Gambar 3.16 Flowchart Melihat Histori Pembacaan Nota pada Aplikasi Android 41](#_Toc135801916)

[Gambar 3.17 Rancangan Tampilan Depan Sistem 42](#_Toc135801917)

[Gambar 3.18 Rancangan Halaman Scan 42](#_Toc135801918)

[Gambar 3.19 Rancangan Hasil Preview Pembacaan 43](#_Toc135801919)

[Gambar 3.20 Rancangan Riwayat Hasil Scan 44](#_Toc135801920)

[Gambar 4.1 Hasil Proses Pembacaan Gambar serta Resize 46](#_Toc135801921)

[Gambar 4.2 Hasil Proses Bluring dan Dilasi pada Gambar 47](#_Toc135801922)

[Gambar 4.3 Hasil Deteksi Tepi Dengan Menggunakan Canny 48](#_Toc135801923)

[Gambar 4.4 Deteksi Kontur Hasil tepi Canny 49](#_Toc135801924)

[Gambar 4.5 Hasil Pemilihan Kontur 50](#_Toc135801925)

[Gambar 4.6 Nota Hasil Segmentasi 51](#_Toc135801926)

[Gambar 4.7 Hasil Deteksi Google Vision 52](#_Toc135801927)

[Gambar 4.8 Gambar Penggunaan Fungsi Annotasi tahap 1 55](#_Toc135801928)

[Gambar 4.9 Gambar Penggunaan Fungsi Annotasi tahap 2 55](#_Toc135801929)

[Gambar 4.10 Hasil Segmentasi Bounding Box 57](#_Toc135801930)

[Gambar 4.11 Hasil Pengolahan Dataset 59](#_Toc135801931)

[Gambar 4.12 Hasil Pengubahan Data Menjadi Encoding 60](#_Toc135801932)

[Gambar 4.13 Hasil Data Latih Classification Report 63](#_Toc135801933)

[Gambar 4.14 Hasil Data Latih Confussion Matrix 63](#_Toc135801934)

[Gambar 4.15 Hasil data uji Classification Report 64](#_Toc135801935)

[Gambar 4.16 Hasil Data Uji Confussion Matrix 64](#_Toc135801936)

[Gambar 4.17 Hasil Web Server Deployment 66](#_Toc135801937)

[Gambar 4.18 Tampilan Home Aplikasi 67](#_Toc135801938)

[Gambar 4.19 Tampilan Menu Sumber Gambar Nota 67](#_Toc135801939)

[Gambar 4.20 Tampilan Proses Cropping Pada Aplikasi 68](#_Toc135801940)

[Gambar 4.21 Tampilan Preview Hasil Pembacaan Nota 68](#_Toc135801941)

[Gambar 4.22 Contoh Pengaruh Pencahayaan Pada Deteksi Sistem 78](#_Toc135801942)

# DAFTAR KODE PROGRAM

[Kode Program 4.1 Pembacaan File Gambar Nota 45](#_Toc132501765)

[Kode Program 4.2 Preprocessing Nota Tahap 1 46](#_Toc132501766)

[Kode Program 4.3 Proses Deteksi Tepi Canny 47](#_Toc132501767)

[Kode Program 4.4 Deteksi Kontur Hasil tepi Canny 49](#_Toc132501768)

[Kode Program 4.5 Pemilihan Kontur 50](#_Toc132501769)

[Kode Program 4.6 Segmentasi Nota 51](#_Toc132501770)

[Kode Program 4.7 Pembacaan Karakter Dengan Google Vision 52](#_Toc132501771)

[Kode Program 4.8 Kode Program Otomasi Pembacaan OCR pada Setiap Nota 53](#_Toc132501772)

[Kode Program 4.9 Fungsi Pembantu Annotasi Dataset 54](#_Toc132501773)

[Kode Program 4.10 Persiapan Dataset 56](#_Toc132501774)

[Kode Program 4.11 Fungsi Normalisasi Dataset 56](#_Toc132501775)

[Kode Program 4.12 Segmentasi Kata Bounding Box 57](#_Toc132501776)

[Kode Program 4.13 Inisiasi AutoProcessor LayoutLM 58](#_Toc132501777)

[Kode Program 4.14 Pengolahan Dataset 58](#_Toc132501778)

[Kode Program 4.15 Pengubahan Data Menjadi Encoding 59](#_Toc132501779)

[Kode Program 4.16 Metrics Pelatihan Model 61](#_Toc132501780)

[Kode Program 4.17 Memuat Model LayoutLM 61](#_Toc132501781)

[Kode Program 4.18 Pelatihan Model 62](#_Toc132501782)

[Kode Program 4.19 Evaluasi Model 62](#_Toc132501783)

[Kode Program 4.20 Web Server Deployment 65](#_Toc132501784)

# DAFTAR TABEL

[Tabel 2.1 State of the Art 6](#_Toc132503431)

[Tabel 2.2 Confusion Matrix 21](#_Toc132503432)

[Tabel 3.1 Spesifikasi Perangkat Keras 24](#_Toc132503433)

[Tabel 3.2 Spesifikasi Perangkat Lunak 25](#_Toc132503434)

[Tabel 3.3 Pelabelan Dataset 30](#_Toc132503435)

[Tabel 3.4 Daftar Sampel *Dataset* Primer 32](#_Toc132503436)

[Tabel 4.1 Contoh Pengujian Inferensi Model 69](#_Toc132503437)

[Tabel 4.2 Rangkuman Hasil Pengujian Inferensi Model 71](#_Toc132503438)

[Tabel 4.3 Contoh Pengujian Kecerahan Nota 75](#_Toc132503439)

[Tabel 4.4 Rangkuman Hasil Pengujian Kecerahan Nota 75](#_Toc132503440)

[Tabel 4.5 Contoh Pengujian Gambar yang Tertutup Bayangan 76](#_Toc132503441)

[Tabel 4.6 Rangkuman Hasil Pengujian Gambar tertutup bayangan 77](#_Toc132503442)

[Tabel 4.7 Tabel Pengujian Variasi Panjang Nota 79](#_Toc132503443)

# 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.

## Latar Belakang

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 belanja pada struk atau nota sebaiknya dilakukan dalam bentuk digital agar tidak mudah hilang. Pemindahan informasi transaksi ke dalam bentuk digital akan memakan banyak waktu jika setiap data transaksi tersebut harus di-*input­* secara manual ke dalam komputer (Kumar, Kaware, & Singh, 2020). Maka dari itu, 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 kalimatpada 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.

Penelitian untuk membuat sistem ekstraksi nota secara otomatis, telah dilakukan dengan menggunakan berbagai metode. Raoui-Outach et al menggunakan lokalisasi dan *Deep Convolution Neural Network* (DCNN) untuk mensegmentasi nama toko. Shi et al dan Lin et al menggunakan *Metode* *Template Matching* berdasarkan *keyword* spesifik pada nota untuk mengekstrak informasi. Meng et al menggunakan model YOLOv3 untuk mensegmentasi area penting pada nota. Namun, sejauh ini sistem masih bergantung pada nota harus memiliki bentuk standar serta bentuk nota yang tidak boleh berubah. Penelitian ini mengusulkan penggunaan model LayoutLM untuk mengenali isi struk dan menggunakan OCR untuk mengekstrak informasi yang terkandung di dalamnya. Sistem yang diusulkan bekerja dengan menggunakan Google Vision OCR untuk mengelompokkan dan mengekstrak kata-kata dalam nota dan menggunakan LayoutLM yang telah dilatih untuk menentukan informasi yang relevan pada nota.

Proses pembacaan nota dengan LayoutLM ini bekerja dengan cara mengakuisisi citra nota yang diambil menggunakan kamera *smartphone* Android. Citra tersebut selanjutnya dikirim menuju *web server* dengan menggunakan sebuah *API Call.* *Web server* akan menerima dan memproses citra nota 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.

## 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 kemampuan sistem dalam membaca variasi nota belanja yang berasal dari toko yang berbeda.

## Tujuan Penelitian

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

1. Menghasilkan sebuah sistem yang dapat membaca dan menyimpan bukti transaksi dalam bentuk digital secara otomatis.

2. Menghasilkan sistem yang dapat mengekstrak informasi pada berbagai variasi nota dengan baik.

## 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.

## 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*.

## 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.

### 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.

### 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.

### Bab III Metodologi Penelitian

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

### 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.

### 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.

# 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.

## 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).

State of the Art

|  |  |  |
| --- | --- | --- |
| **No** | **Jurnal** | **Keterangan** |
| 1 | **Judul**  *Universal Language Model Fine-tuning for Text Classification*  **Peneliti**  Howard, Jeremy  Ruder, Sebastian  **Tahun**  2018  **Publikasi**  *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* | 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 Modelbernama *Universal Language* Model *Fine-tuning* (ULMFiT). ULMFiT adalah sebuah Model yang dirancang dan dikembangkan dengan tujuan agar proses *Transfer Learning* 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). |
| 2 | **Judul**  *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*  **Peneliti**  Devlin, Jacob  Chang, Ming-Wei  Lee, Kenton  Toutanova, Kristina  **Tahun**  2019  **Publikasi**  *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1* *(Long and Short Papers)* | Devlin melanjutkan perkembangan penggunaan *transfer learning* pada permasalahan NLP dan membuat sebuah Modelbernama *Bidirectional Encoder Representations from Transformers* (BERT). BERT adalah sebuah *pretrained* Model yang menggunakan representasi *Deep Bidirectional* dari teks dengan cara menggabungkan konteks kiri dan kanan dari teks tersebut pada setiap *layer*-nya. Hasil pembuatan Model BERT ini berhasil mendapatkan akurasi di atas 82.1% yang berhasil melampaui Model SOTA dalam berbagai *dataset*. Model BERT yang dihasilkan juga dapat dengan mudah di-*tuning* untuk menyelesaikan berbagai jenis permasalahan NLP seperti *language inference*, dan *question answering* tanpa mengubah banyak dari arsitektur Model (Devlin et al., 2018). |
| 3 | **Judul**  *BioBERT: a pre-trained biomedical language representation model for biomedical text mining*  **Peneliti**  Jinhyuk Lee  Wonjin Yoon  Sungdong Kim  Donghyeon Kim  Sunkyu Kim,  Chan Ho So,  Jaewoo Kang  **Tahun**  2019  **Publikasi**  *Bioinformatics, Volume 36, Issue 4, February 2020* | Penelitian tentang *Finetuning* dari ModelBERT, 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 *tools* yang akurat. Hasil penerapan *Finetuning* pada Model BERT berhasil membuat model bernama BioBERT dan mendapatkan akurasi mencapai 72 % dalam berbagai *dataset* serta berhasil melampaui Model *state of the art* dalam beberapa kategori ekstraksi informasi biomedis (Lee et al., 2020). |
| 4 | **Judul**  *Automatic Receipt Recognition System Based on ArtificialIntelligence Technology*  **Peneliti**  Lin, Cheng Jian  Liu, Yu Cheng  Lee, Chin Ling  **Tahun**  2021  **Publikasi**  *Applied Sciences, Volume 12, Issue 2* *(January-2 2022)* | Metode OCR pada nota belanja pernah dilakukan dengan mengaplikasikan *Template Matching* pada nota yang ingin dibaca. Hasil dari penggunaan *Metode* *Template Matching* tersebut selanjutnya akan dibaca dengan menggunakan Model *Deep Learning Yolov4-s*. Sistem yang dikembangkan ini berhasil mendapatkan akurasi sebesar 80.93% dengan percobaan menggunakan CNN dan sebesar 99.39% dengan menggunakan *Yolov4-s* yang dikembangkan dalam pembacaan karakter (Lin et al., 2022). |
| 5 | **Judul**  Implementasi Tesseract OCR untuk Pembuatan Aplikasi Pengenalan Nota pada Android  **Peneliti**  Andreas, Yoel  Gunadi, Kartika  Purbowo, Anita Nathania  **Tahun**  2020  **Publikasi**  Jurnal Infra (2020) vol 8 | Pembacaan nota pernah dilakukan dengan memanfaatkan *Library* *BlinkReceipt*. *BlinkReceipt* merupakan sebuah *Application Programming Interface* (API)yang dapat digunakan dalam platform *IOS*, *Android*, serta melalui *javascript*. *Library* *BlinkReceipt* adalah sebuah *library* yang dapat digunakan secara khusus untuk membaca nota belanja. Hasil yang didapatkan dalam penggunaan *Library BlinkReceipt* 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). |
| 6 | **Judul**  *Deep Learning For Automatic Sale Receipt Understanding*  **Peneliti**  Raoui-Outach, Rizlene  Million-Rousseau  Benoit, Alexandre  Lambert, Patrick  **Tahun**  2017  **Publikasi**  *Proceedings of the 7th International Conference on Image Processing Theory, Tools and Applications, IPTA 2017* | 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 dari merek toko dilakukan dengan *Deep Convolutional Neural Networks* (CNN). Hasil dari percobaan ini berhasil melokalisasi merek toko sebesar 98,7% (Raoui-Outach et al., 2018). |
| 7 | **Judul**  Pemanfaatan *Optical Character Recognition* dan *Text Feature Extraction* untuk Membangun Basis Data Pengaduan Tenaga Kerja  **Peneliti**  Puspitarani, Yan  Syukriyah, Yenie  **Tahun**  2020  **Publikasi**  JURNAL RESTI Vol. 4 No. 4 (2020) | Pembacaan pengaduan ketenagakerjaan dengan menggunakan *Tesseract OCR* dan *NTLK toolkit* pernah dilakukan untuk mengekstrak informasi pada surat*.* Penelitian ini menggunakan *NTLK toolkit* untuk memahami isi surat, sedangkan proses ekstraksi tulisan surat dilakukan dengan menggunakan *Tesseract OCR*. Penelitian ini berhasil mengekstrak informasi sebesar 66.7% pada surat tulisan tangan dan sebesar 91,67% pada surat yang diketik (Puspitarani & Syukriyah, 2020). |
| 8 | **Judul**  *Segmentation for document layout analysis: not dead yet*  **Peneliti**  Markewich, Logan  Zhang, Hao  Xing, Yubin  Lambert-Shirzad, Navid  Jiang, Zhexin  Lee, Roy Ka-Wei  Li, Zhi  Ko, Seok-Bum  **Tahun**  2020  **Publikasi**  *International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25* | Penggunaan *Metode* *Weighted Bounding Box Regression Loss* 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 *Dense Article Dataset* (DAD) dan *dataset* *PubLayNet* memiliki f1 *score* sebesar 96.26% dan 97.11% dengan menggunakan Model *DeeplabV3+* (Markewich et al., 2022). |
| 9 | **Judul**  *TableSegNet: a fully convolutional network for table detection and segmentation in document images*  **Peneliti**  Nguyen, Duc-Dung  **Tahun**  2022  **Publikasi**  *International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25* | Model *TableSegNet* merupakan Model segmentasi yang dapat mendeteksi tabel dalam sebuah gambar dokumen. *TableSegNet* menggunakan arsitektur *fully convolutional network* untuk mendeteksi dan membedakan tabel secara bersamaan. Hasilnya penelitian Model segmentasi *TableSegNet* ini dapat menghasilkan akurasi sebesar 90% dalam dataset ICDAR2019 (Nguyen, 2022). |
| 10 | **Judul**  *MRZ code extraction from visa and passport documents using convolutional neural networks*  **Peneliti**  Liu, Yichuan  James, Hailey  Gupta, Otkrist  Raviv, Dan  **Tahun**  2022  **Publikasi**  *International Journal on Document Analysis and Recognition (IJDAR) (2022) vol 25* | Ekstraksi kode MRZ pada dokumen *visa* dan *passport* pernah dilakukan dengan menggunakan *Convolutional Neural Networks* (CNN). Model CNN dibuat untuk mendeteksi *MRZ code* dari gambar *passport* digital. Hasilnya Model ini dapat mendeteksi 100% dari kode MRZ dan 99.25% *macro-f1* dari pengenalan karakter pada *dataset* *passport* dan Visa (Liu et al., 2022). |
| 11 | **Judul**  *Learning from similarity and information extraction from structured documents*  **Peneliti**  Holeček, Martin  **Tahun**  2021  **Publikasi**  *International Journal on Document Analysis and Recognition (IJDAR) (2021) vol 24* | Penelitian tentang ekstraksi informasi pada dokumen yang terstruktur pernah dilakukan dengan menggunakan arsitektur *Siamese Networks*. Penggunaan arsitektur *Siamese Networks* dikombinasikan dengan *Metode* *Similarity*, *One-Shot Learning*, dan *Context/Memory Awareness* untuk melakukan proses ekstraksi informasi pada dokumen. Hasil dari arsitektur pada penelitian ini berhasil meningkatkan skor f1 sebesar 8.25% bila dibandingkan dengan arsitektur *query answer* (Holeček, 2021)*.* |
| 12 | **Judul**  *Implementation of Zoning and K-Nearest Neighbor in Character Recognition of Wrésastra Script*  **Peneliti**  Darma, I Wayan Agus Surya **Tahun**  2019  **Publikasi**  Lontar Komputer : Jurnal Ilmiah Teknologi Informasi (2019) | Penelitian tentang pembacaan karakter huruf Bali pada *wrésastra script* dan karakter huruf Bali pernah dilakukan dengan menggunakan *Zoning Feature Extraction* serta K-*Nearest Neighbor* (KNN). Penggunaan *Zoning Feature Extraction* dilakukan untuk mengekstrak fitur dari setiap karakter dan menjadikannya *embedding* untuk diklasifikasikan dengan ModelKNN. Hasil yang pada penelitian ini adalah Modelterbaik didapatkan saat parameter *neighbor* yang digunakan adalah tiga dengan akurasi mencapai 97.5% (Darma, 2019). |
| 13 | **Judul**  *Handwriting Identification Using Deep Convolutional Neural Network Method* **Peneliti**  Sudana, Oka  Gunaya, I. Wayan  Putra, I. Ketut Gede Darma  **Tahun**  2020  **Publikasi**  *Telecommunication Computing Electronics and Control (2020) vol 18* | Penelitian tentang klasifikasi karakter penulisan tangan pernah dilakukan dengan menggunakan *Convolutional Neural Network* (CNN). Arsitektur CNN digunakan untuk mengklasifikasi tulisan tangan adalah VGG19 dengan layer *output* yang telah dimodifikasi. Hasil dari penelitian ini mendapatkan 90% akurasi dengan menggunakan gambar *grayscale* dan berhasil mengklasifikasi karakter penulisan orang dengan benar (Sudana et al., 2020). |
| 14 | **Judul**  *Detection of fake news using deep learning CNN–RNN based methods*  **Peneliti**  Sastrawan, I. Kadek  Bayupati, I. P.A.  Arsa, Dewa Made Sri  **Tahun**  2020  **Publikasi**  ICT Express (2022) vol 8 | Penelitian tentang deteksi berita bohong pernah dilakukan dengan menggunakan Model CNN, *Bidirectional* LSTM, dan *ResNet*. Proses pengubahan kata menjadi *token* pada penelitian ini dilakukan dengan menggunakan *embedding Word2Vec*, *GloVe*, dan *FastText*. Hasil dari penelitian ini mendapatkan kombinasi *embedding* *GloVe* dengan arsitektur *Bidirectional* LSTM memberikan hasil yang terbaik dengan akurasi melampaui 94.6%.(Sastrawan et al., 2022). |
| 15 | **Judul**  *Balinese Script Recognition Using Tesseract Mobile Framework*  **Peneliti**  Indrawan, Gede  Asroni, Ahmad  Joni Erawati Dewi, Luh  Gunadi, I Gede Aris  Paramarta, I Ketut  **Tahun**  2020  **Publikasi**  Lontar Komputer : Jurnal Ilmiah Teknologi Informasi (2022) vol 13 | Penelitian tentang pengenalan karakter bahasa Bali pernah dilakukan dengan menggunakan *Tesseract OCR* 5. Pelatihan Model *Tesseract OCR* 5 dilakukan menggunakan *tools* pelatihan *Tesseract OCR*. Penggunaan *tools* pelatihan *Tesseract OCR* dilakukan untuk mengekstrak dan mengenali karakter bahasa Bali yang terdeteksi pada sistem.Penelitian ini berhasil mendapatkan skor *coincidence* sebesar 66.67% dengan memperhitungkan hierarki dari masing-masing karakter, kata, kalimat, serta paragraf dari penulisan bahasa Bali (Indrawan et al., 2022). |
| 16 | **Judul**  *Research on a handwritten character recognition algorithm based on an extended nonlinear kernel residual network*  **Peneliti**  Zheheng Rao  Chunyan Zeng  Minghu Wu  Zhifeng Wang  Nan Zhao  Min Liu  Xiangkui Wan  **Tahun**  2018  **Publikasi**  *KSII Transactions on Internet and Information Systems* (2018) vol 12 | Penelitian tentang pengenalan tulisan tangan ini dilakukan untuk meneliti sebuah metode yang dapat mengurangi proses komputasi model OCR. Penelitian ini memberikan pernyataan bahwa model *deep learning* yang menggunakan *extremely deep network* dalam melakukan OCR memerlukan kemampuan komputasi yang tinggi. Solusi yang diberikan oleh penelitian ini untuk mempercepat proses pelatihan model adalah dengan menggunakan *Extended Nonlinear Kernel Residual Network*. Hasil yang didapatkan adalah model dapat mencapai pengenalan karakter hingga 97.72% dengan waktu latih yang lebih singkat dari model berbasis *Residual Network* maupun model berbasis CNN (Rao et al., 2018). |
| 17 | **Judul**  *Segmentation based, omnifont printed Arabic character recognitionwithout font identification*  **Peneliti**  Aziz Qaroush  Abdalkarim Awad  Mohammad Modallal  Malik Ziq  **Tahun**  2022  **Publikasi**  *Journal of King Saud University - Computer and Information Sciences* (2022) vol 34 | Penelitian ini memberikan sebuah metode untuk melakukan pengenalan karakter tanpa menggunakan proses pengenalan *font* dari karakter. Pendekatan yang dilakukan adalah dengan menggunakan *indirect character segmentation*. Hasil akurasi rata-rata yang didapatkan dari penelitian ini adalah 95% tanpa pengenalan tipe *font* ataupun teknik *post processing* lainnya (Qaroush et al., 2022). |
| 18 | **Judul**  *IRIS: Smart Phone Aided IntelligentReimbursement System using DeepLearning* **Peneliti**  Meng, Yang  Wang, Run  Wang, Juan  Yang, Jie  Gui, Guan  **Tahun**  2022  **Publikasi**  IEEE Access (2019) vol 7 | Penelitian milik Meng et al menggunakan sebuah model YOLOv3 yang dilatih untuk dapat melakukan segmentasi area penting dalam sebuah nota. Area-area penting ini kemudian akan dibaca dengan menggunakan OCR. Penggunaan *Metode* *Area Segmentation* untuk membaca nota memiliki kekurangan yaitu model YOLOv3 harus mengenali bentuk nota yang ingin dibaca. Akurasi OCR pada penelitian ini setelah informasi berhasil disegmentasi dengan baik mencapai 96,2% hingga 99,1% (Meng et al., 2019). |

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.

## 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).

## 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).

## 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).

## 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.

## 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, questionaire, resume*, dll (Xu et al., 2020).

|  |
| --- |
|  |

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 Modelini.

|  |
| --- |
|  |

Input Embedding Model LayoutLM

Gambar 2.2 merupakan ilustrasi pembuatan *input embedding* yang digunakan untuk melatih Model LayoutLM. Model LayoutLM memerlukan sebuah gambar, kumpulan kata, beserta *bounding box*, dalam bentuk *embedding* sebagai *input* pelatihan Model. Proses pengubahan data menjadi *embedding* dapat dilakukan menggunakan *Auto Processor* yang disediakan oleh library *Transformer*.

*Auto Processor* digunakan untuk mempermudah dan menyamakan data latih dengan format awal pelatihan model. *Auto Processor* akan menerapkan berbagai proses pada data *input* seperti: *resize* gambar, *sentence* *padding*, *character* *handling*, *attention mask generation*, dll. Penggunaan *Auto Processor* mempercepat proses penyiapan *dataset* sehingga proses pelatihan dapat berjalan lebih cepat.

## Metriks Evaluasi Model

Evaluasi pada Model dilakukan menggunakan *confussion matrix* serta nilai akurasi, presisi, *recall*, dan *f1-score* yang didapatkan Model pada data uji. *Contussion 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).

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.

1. *True Positive* (TP) merupakan label positif yang berhasil diprediksi positif
2. *False Positive* (FP) merupakan label negatif yang salah diprediksi sebagai positif
3. *False Negative* (FN) merupakan label positif yang salah diprediksi negatif
4. *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.

|  |  |
| --- | --- |
|  | (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.

|  |  |
| --- | --- |
|  | (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.

|  |  |
| --- | --- |
|  | (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.

|  |  |
| --- | --- |
|  | (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.

# 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.

## 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.

## 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.

### 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.

### 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).

## 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.

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.

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.

## 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.

|  |
| --- |
|  |

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.

### 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.

|  |
| --- |
|  |

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.

#### 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.

|  |
| --- |
|  |

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 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*.

#### 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 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*.

#### 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*.

Pelabelan Dataset

|  |  |
| --- | --- |
| **Nama Label** | **Kode Label** |
| *Ignore* | 0 |
| *Store\_name\_value* | 1 |
| *Date\_value* | 2 |
| *Time\_value* | 3 |
| *Prod\_item\_key* | 4 |
| *Prod\_item\_value* | 5 |
| *Prod\_quantity\_key* | 6 |
| *Prod\_quantity\_value* | 7 |
| *Prod\_price\_key* | 8 |
| **Nama Label** | **Kode Label** |
| *Prod\_price\_value* | 9 |
| *Subtotal\_key* | 10 |
| *Subtotal\_value* | 11 |
| *Total\_key* | 12 |
| *Total\_value* | 13 |
| *Others* | 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 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.

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.

### Finetuning Model LayoutLM

Model *Finetuning* adalah sebuah istilah dalam *machine learning* untuk menyelesaikan permasalahan *Natural Language Processing* (NLP). *Finetuning* merupakan proses yang menyerupai *transter 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.

|  |
| --- |
|  |

Flowchart Pelatihan LayoutLM

Gambar 3.7 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.

### Percobaan Inferensi Model

Percobaan inferensi model menguji hasil deteksi model yang telah dilatih dalam membaca satu buah nota. Proses ini memperlihatkan alur deteksi nota, mulai dari proses memuat nota, hingga hasil deteksi sistem.

|  |
| --- |
|  |

Percobaan Inferensi Model

Gambar 3.8 menunjukkan proses *preprocessing* yang dilakukan pada gambar nota. Bagian kiri gambar menunjukkan gambar nota asli, dan bagian kanan gambar menunjukkan gambar hasil *contrast stretching*.

|  |
| --- |
|  |

Hasil Preprocessing Inferensi Nota

Gambar 3.9 merupakan proses pembacaan OCR menggunakan *Layout Parser* dan *Google Vision API*. Hasil dari proses ini memberikan kumpulan kata beserta *bounding box* yang digunakan sebagai *input* pada inferensi LayoutLM.

|  |
| --- |
|  |

Hasil Inferensi LayoutLM

Gambar 3.10 merupakan hasil prediksi dari model LayoutLM yang digunakan. Model akan memberikan sebuah *array* label dalam bentuk *torch* sesuai dengan jumlah kata/*bounding box* yang telah terdeteksi sebelumnya.

|  |
| --- |
|  |

Hasil Deteksi Sistem

Gambar 3.11 merupakan hasil akhir ketika prediksi label dari Model LayoutLM ditampilkan pada gambar nota hasil segmentasi. Hasil deteksi ini kemudian di-*filter* untuk menampilkan informasi penting pada nota.

### 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.

#### Confussion Matrix

Tahap penilaian *confussion matrtix* 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*.

#### 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*.

### Skenario 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.

## 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.

|  |
| --- |
|  |

Gambaran Umum Sistem

Gambar 3.12 merupakan gambaran umumdari 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.

## 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*.

|  |
| --- |
|  |

Flowchart Pembacaan Nota pada Aplikasi Android

Gambar 3.13 merupakan *flowchart* alur pembacaan nota yang terdapat pada aplikasi Android. Pembacaannota 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.

|  |
| --- |
|  |

Flowchart Proses Pembacaan Nota pada Web Server

Gambar 3.14 merupakan gambaran *flowchart* pemrosesanpada *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* notaserta 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.

|  |
| --- |
|  |

Flowchart Preprocessing Gambar pada Server

Gambar 3.15 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*.

|  |
| --- |
|  |

Flowchart Melihat Histori Pembacaan Nota pada Aplikasi Android

Gambar 3.16 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.

## 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.

|  |
| --- |
|  |

Rancangan Tampilan Depan Sistem

Gambar 3.17 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.

|  |
| --- |
|  |

Rancangan Halaman Scan

Gambar 3.18 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*.

|  |
| --- |
|  |

Rancangan Hasil Preview Pembacaan

Gambar 3.19 adalah gambar rancangan halaman *preview* setelah didapatkan hasil pembacaan oleh *web server*. Halaman ini 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*.

|  |
| --- |
|  |

Rancangan Riwayat Hasil Scan

Gambar 3.20 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.

# 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.

## 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*.

### 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() |

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*.

|  |
| --- |
|  |

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() |

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*.

|  |
| --- |
|  |

Hasil Proses Bluring 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() |

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.

|  |
| --- |
|  |

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\_conturs\_m = cv2.drawContours(img\_rezise.copy(), contours\_m, -1, (0,255,0), 3)  all\_conturs\_b = cv2.drawContours(img\_rezise.copy(), contours\_b, -1, (0,255,0), 3)  plt.figure(figsize=(15,15))  plt.subplot(1,2,1)  plt.imshow(all\_conturs\_m)  plt.subplot(1,2,2)  plt.imshow(all\_conturs\_b)  plt.show() |

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.

|  |
| --- |
|  |

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\_rezise.copy(), largest\_contours, -1, (0,255,0), 3)  plt.figure(figsize=(10,10))  plt.imshow(image\_with\_largest\_contours)  plt.show() |

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.

|  |
| --- |
|  |

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\_rezise.copy(), [receipt\_contour], -1, (0, 255, 0), 2)  result = wrap\_perspective(img\_rezise.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\_rezise[y:y+h, x:x+w]  plt.figure(figsize=(10,10))  plt.imshow(result)  plt.show() |

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.

|  |
| --- |
|  |

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*.

### 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) |

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.

|  |
| --- |
|  |

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 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.

### 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 3.3 yang dijelaskan dalam metodologi penelitian.

|  |
| --- |
| 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) |

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 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 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.

## Finetuning Model LayoutLM

Proses *Finetuning* dilakukan dengan menggunakan data latih sebanyak 80 dari 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.

### 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) |

Persiapan Dataset

Kode program 4.10 adalah kode program yang digunakan untuk memuat data sekunder *WildReceipt*. Data dimuat menggunakan fungsi load\_dataset, kemudian Informasi kata, *bounding box*, serta label diekstrak ke dalam variabel.

|  |
| --- |
| 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),  ] |

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 |

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 segmentasisetelah diterapkan normalisasi *bounding box*.

|  |
| --- |
|  |

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.

### 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) |

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) |

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.

|  |
| --- |
|  |

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 ModelLayoutLM.

|  |
| --- |
| 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") |

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 *encoding* memiliki format *torch* dengan jumlah data latih sebanyak 1267 dan data uji berjumlah 472.

|  |
| --- |
|  |

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*.

### 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("seqeval")  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"],  } |

Metrik 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) |

Memuat Model LayoutLM

Kode Program 4.17 adalah kode program yang digunakan untuk memuat Model LayoutLM dasar. Kode ini memuat ModelLayoutLM 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() |

Pelatihan Model

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

### 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]) |

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.

|  |
| --- |
|  |

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.

|  |
| --- |
|  |

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.

|  |
| --- |
|  |

Hasil data uji Classification Report

Gambar 4.15 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.

|  |
| --- |
|  |

Hasil Data Uji Confussion Matrix

Gambar 4.16 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*.

## 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.

### 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* Modelpada 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)  bytearrimg = decodeB64(data)  byteimg = bytearray(bytearrimg)  pil\_image = Image.open(io.BytesIO(byteimg))  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 successResponse(singleReceipt(img\_info),img\_str,"success")  except Exception as e:  return badRequest(e,"error")  if \_\_name\_\_ == '\_\_main\_\_':  app.run(debug = True) |

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.

|  |
| --- |
|  |

Hasil Web Server Deployment

Gambar 4.17 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.

### 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.

|  |
| --- |
|  |

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.

|  |
| --- |
|  |

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*.

|  |
| --- |
|  |

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.

|  |
| --- |
|  |

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.

## 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

### 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.

Contoh Pengujian Inferensi Model

|  |  |
| --- | --- |
|  | 1. Nama Toko terbaca dengan tepat  * Terdapat salah deteksi  1. Tanggal terbaca dengan tepat 2. Waktu terbaca dengan tepat 3. Produk dapat terbaca sebagian  * 1 kuantitas tidak terbaca  1. Total terbaca dengan tepat   Akurasi : 85,7% |
|  | 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   Akurasi : 93,7% |
|  | 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian  * 2 harga tidak terdeteksi * 1 nama produk tidak terdeteksi  1. Total terbaca dengan benar   Akurasi : 80% |
|  | 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat  * 1 kuantitas salah terdeteksi  1. Total terbaca dengan tepat  * 1 total salah terdeteksi   Akurasi : 80% |
|  | 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   Akurasi : 100% |
|  | 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   Akurasi : 100% |

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.

Rangkuman Hasil Pengujian Inferensi Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Nama Toko | Jumlah Uji | Hasil Uji | Akurasi Rata-Rata |
| 1 | Indomaret | 5 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 91,7% |
| 2 | Circle K | 3 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 93,3% |
| 3 | McDonald | 3 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 86,9% |
| 4 | Twisterdog | 3 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 90,9% |
| 5 | Alfamart | 2 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 Terdeteksi | | Salah Deteksi | | 2/2 | | 0/2 | | 0/2 | | 0 | | 100% |
| 6 | Mixue | 2 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 94,4% |
| 7 | Lainnya | 14 | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | Nama Toko | | | | | | | | 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 | | 88,3% |

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.

### 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. Proses pengujian variasi kecerahan dibagi menjadi dua. Pengujian pertama dilakukan untuk menentukan pengaruh kecerahan gambar nota pada kemampuan deteksi model. Metode pengujian kedua dilakukan untuk menentukan pengaruh bayangan objek yang terdapat pada gambar nota saat proses pengambilan gambar.

Contoh Pengujian Kecerahan Nota

|  |  |
| --- | --- |
|  | 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat  * 2 kuantitas terbaca sebagian  1. Total terbaca dengan tepat   Akurasi : 89,4% |
|  | 1. Nama Toko tidak dapat terbaca 2. Tanggal tidak dapat terbaca 3. Waktu tidak dapat terbaca 4. Produk terbaca sebagian  * Kelima kuantitas produk tidak terdeteksi  1. Total tidak dapat terbaca   Akurasi : 73,6% |

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.

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 |
| **NO** | **Nama Toko** | **Label** | **Terang** | **Gelap** |
| 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.

Contoh Pengujian Gambar yang Tertutup Bayangan

|  |  |
| --- | --- |
|  | 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   Akurasi : 92,8% |
|  | 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian   • Ketiga kuantitas tidak terbaca   1. Total terbaca dengan tepat   Akurasi : 71,4% |
|  | 1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian  * satu kuantitas produk tidak terdeteksi  1. Total terbaca dengan tepat   Akurasi : 85,7% |

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.

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 |
|  |  |  |  |  |  |
| **NO** | **Nama Toko** | **Label** | **Tertutup Bayangan**  **< 30%** | **Tertutup Bayangan**  **50%** | **Tertutup Bayangan**  **> 80%** |
| 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.

|  |
| --- |
|  |

Contoh Pengaruh Pencahayaan 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.

### 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 Pengujian Variasi Panjang Nota

|  |  |
| --- | --- |
|  | Ukuran nota :   * Panjang : 3722px * Lebar : 504px   Jumlah Produk : 31 item   1. Nama Toko tidak terbaca 2. Tanggal tidak terbaca 3. Waktu tidak terbaca 4. Produk tidak terbaca  * 21 produk terbaca sebagian * 10 produk tidak terdeteksi  1. Total tidak terbaca |
|  | Ukuran nota :   * Panjang : 3110px * Lebar : 503px   Jumlah Produk : 28 item   1. Nama Toko terbaca dengan tepat 2. Tanggal tidak terbaca 3. Waktu tidak terbaca 4. Produk terbaca sebagian  * 20 produk terbaca dengan tepat * 8 produk terbaca sebagian  1. Total tidak terbaca |
|  | Ukuran nota :   * Panjang : 2213px * Lebar : 679px   Jumlah Produk : 20 item   1. Nama Toko tidak terbaca 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca sebagian  * 17 produk terbaca sebagian * 3 produk tidak terdeteksi  1. Total tidak terbaca |
|  | Ukuran nota :   * Panjang : 2334px * Lebar : 974px   Jumlah Produk : 10 item   1. Nama Toko terbaca dengan tepat 2. Tanggal terbaca dengan tepat 3. Waktu terbaca dengan tepat 4. Produk terbaca dengan tepat  * 1 produk terbaca sebagian  1. Total terbaca dengan tepat |

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.

### Hasil Pengujian Sistem

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).

Berdasarkan ketiga pengujian yang dilakukan, akurasi model yang didapatkan ketika proses evaluasi awal yaitu 97,98% mengalami fluktuasi dalam pengujian secara *realtime*. Model masih belum dapat bekerja dengan optimal pada beberapa variasi nota yang memiliki letak informasi yang berdekatan serta *font* yang terlalu kecil. Variasi ini membuat akurasi pengujian sistem menurun ke angka 90%. 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.

### 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.

#### 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.

#### 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.

## Perbandingan Hasil

Perbandingan hasil digunakan untuk membandingkan metode yang digunakan dalam penelitian ini dengan hasil yang didapatkan dalam penelitian yang serupa. Penelitian yang digunakan sebagai perbandingan adalah penggunaan *Metode* *Deep Convolution Neural Network* (DCNN) untuk segmentasi dan lokalisasi nama toko milik Raoui-Outach et al, *Metode* *Template Matching* milik Shi et al dan Lin et al serta *Metode* *Area Segmentation* milik Meng et al.

Hasil dari penggunaan metode yang diajukan pada penelitian ini adalah informasi penting pada nota dapat diekstrak dengan baik tanpa perlu mengetahui *template* dari nota yang ingin dikenali, dengan akurasi rata-rata 90%. Sedangkan keempat penelitian terkait, harus mengenali nota yang ingin dibaca dengan menggunakan informasi yang sudah tersimpan pada sistem, seperti : letak *keyword*, *template* nota yang sama, bentuk nota yang ter-standardisasi, dll.

Perbandingan penggunaan metode yang diajukan dengan metode pada penelitian terkait adalah pada keempat penelitian tersebut, pengenalan nota bersifat spesifik, sedangkan penelitian ini mengenali nota secara general. Pengenalan nota secara general dapat membaca lebih banyak variasi nota. Namun, pengenalan secara general berpotensi membuat beberapa bentuk nota yang unik tidak dapat dikenali dengan baik oleh sistem. Sebaliknya pengenalan nota secara spesifik dapat membandingkan bentuk nota yang dibaca dengan data yang telah tersimpan, sehingga pembacaan dapat dilakukan dengan lebih baik. Namun, nota yang dibaca harus dikenali oleh sistem.

# PENUTUP

## 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 pembaca nota secara otomatis, berhasil membaca berbagai variasi nota dengan akurasi rata-rata sistem sebesar 90%. Beberapa permasalahan yang menyebabkan menurunnya akurasi sistem berasal dari variasi nota seperti : panjang nota, kecerahan gambar nota, variasi letak informasi penting, format penulisan barang, variasi nama indikator total belanja, dll.

## 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*.

# 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, G., Asroni, A., Joni Erawati Dewi, L., Gunadi, I. G. A., & Paramarta, I. K. (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., 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

Meng, Y., Wang, R., Wang, J., Yang, J., & Gui, G. (2019). IRIS: Smart Phone Aided Intelligent Reimbursement System Using Deep Learning. *IEEE Access*, *7*, 165635–165645. https://doi.org/10.1109/ACCESS.2019.2953501

Ngurah, G., Riantama, S., Piarsa, N., Made, G., & Sasmita, A. (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, D., Rahmaastri, Della Anggi, Setiawan, K., Suryani, D., & Purnama, Y. (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.icte.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

Shi, S., Cui, C., & Xiao, Y. (2020). An Invoice Recognition System Using Deep Learning. *2020 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, 416–423. https://doi.org/10.1109/ICICAS51530.2020.00093

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