

Skanner for kvitteringer

Sivert M. Skarning

Mars 2019

1 Introduksjon

Som et prosjekt i faget bildebehandling og mønstergjenkjenning ved Høgskolen i Østfold har jeg valgt å utvikle en digital kvitteringskanner. En kvitteringskanner er et program som ved hjelp av et kamera klarer å konvertere en kvittering fra fysisk til digital form. I figur 1 ser vi et utdrag fra en kvittering.



Figure 1: Eksempel på en kvittering fra en svensk butikk

2 Oppgave

I dette prosjektet vil vi kun se på teknologien bak analysen av bilde. Det vil ikke inneholde beskrivelse av hva som skjer før og etter denne analysen, som hvordan man får inn bilde som input eller hvordan teksten blir fremvist. Oppgaven blir dermed å få inn et bilde av en kvittering finne ut hva som har blitt kjøpt,

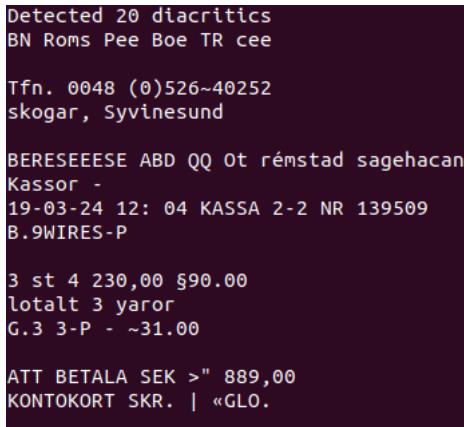
prisen, hvor produktet er blitt kjøpt, avslag og eventuelle skatter som har blitt betalt på produktet. Prosjektet kommer til å bruke kodebibliotek og eksisterende teknologier, men egen programkode skal styre retting og behandling av bilde.

3 Teknologi

Det finnes mange teknologier innenfor bildebehandling og tekstgjenkjenning. Jeg vil i dette prosjektet fokusere på å velge det som er enklest og implementere og open-source programvare der det er tilgjengelig.

3.0.1 Tekstgjenkjenning

Tesseract er et open-source programvare som utfører tekstgjenkjenning. Den kan utfører tekstgjennkjenning på mange typer filformater. Den har også API'er som gjør at det er mulig å bruke det med Python og andre programmeringspråk¹. I figur 2 ser dere output som blir generert etter å ha skannet bilde i figur 3



The image shows the output of Tesseract OCR on a scanned receipt. The text is white on a dark background. It includes:

- Detected 20 diacritics
- BN Roms Pee Boe TR cee
- Tfn. 0048 (0)526~40252
- skogar, Syvinesund
- BERESEESE ABD QQ Ot rémstad sagehacan
- Kassor -
- 19-03-24 12: 04 KASSA 2-2 NR 139509
- B.9WIRES-P
- 3 st 4 230,00 §90,00
- totalt 3 yaror
- G.3 3-P - ~31.00
- ATT BETALA SEK >" 889,00
- KONTOKORT SKR. | «GLO.

Figure 2: Output generert av Tesseract

Deskewing For at bilde skal kunne bli analysert på riktig måte, er vi nødt til skalere, kroppe og rotere bilde. Rotering av bilde kan bli gjort med et konsept som kalles homography. Her kan man bruke referansepunkter på et annet bilde til å rette det. Skalering og kropping er det mulig å programmere selv, eller bruke innebygde funksjoner i scikit-image.²

¹<https://www.learnopencv.com/deep-learning-based-text-recognition-ocr-using-tesseract-and-opencv/>

²<https://www.learnopencv.com/image-alignment-feature-based-using-opencv-c-python/>

Oppskarpning Det kan være utfordrende for tekstgjenkjennings-algoritmer å kjenne igjen tekst. De fleste programmer som kjører tekstgjenkjenning kjører egne filtre for å optimalisere det for tekstgjenkjenning. For å få mer kontroll og øke sjansen for å at gjenkjenningen blir nøyaktig vil jeg bruke min egen optimerer. Den skal kunne gjøre bilde monokromt, fjerne støy, gjøre bilde skarpere og forbedre kontrasten.³ ⁴



Figure 3: Behandlet versjon av figur 1

4 Gjennomføring

4.1 Deskewing

Et problem med å gjennomføre tekstgjenkjenning på et bilde uten å ha behandlet det først er at man ikke har noen garanti på at bilde har riktig vinkel. Hvis teksten ikke går parallelt med bilde vil resultatet bli dårligere. Som observert i figur 5 og figur 6 har vi et bilde som er riktig orientert og et bilde som er feil orientert. På bildet som er feil orientert klarer ikke tesseract å gjenkjenne teksten. På bildet som er riktig orientert får vi følgende resultat som sett i figur 4. Dette resultatet er bedre, men fortsatt langt fra treffsikkert.

Hough

Linjeanalyse

Homography

³<https://docparser.com/blog/improve-ocr/>

⁴<https://docparser.com/blog/improve-ocr-accuracy/>

BPYSOIK :PIS80d
YAN SZ1 000 196
ve NN ante Der PG MA NE
LH0X
JaJ8A 7 UNS
pr 139nGeg
1 20 oBueu 114008 BNDRUOG
Sy - SLI :14PSUY
g500679t :1998
gereZerst :SUPII
LL :67:560 610260 90
: JUB I
uap|eH ZGA| USUNSY
P[OL1S0 | Japeud SIES] USNNIS
x BULJØLAYSLG *
overs 19206 varnS set Bl EP VAA DOP OG MAE EY sten er

Figure 4: Resultate av tessearct OCR



Figure 5: Bilde med feil rotering



Figure 6: Bilde med riktig rotering