Látórendszer fejlesztése DBC-k termikus tranziens mérésautomatizálásához teljesen konvolúciós mélytanulásos neurális hálók, képfeldolgozó algoritmusok és Fa-struktúrájú Parzen Becslő használatával

Vision system development for thermal transient measurement automation of DBCS using fully convolutional deep learning neural networks, image processing algorithms and tree-structured parzen estimator

BÉKÉSI Gergő Bendegúz[1], Dr. EKLER Péter[1], Dr. URBIN Ágnes[2], SZŐKE Szilárd[3]

[1]Budapesti Műszaki és Gazdaságtudományi Egyetem, Automatizálás és Alkalmazott Informatika Tanszék, Budapest, Magyar Tudósok Körútja 2, 1117., (06 1) 463 2870, bekesigergobendeguz@edu.bme.hu, ekler.peter@aut.bme.hu, https://www.aut.bme.hu/
[2]Budapesti Műszaki és Gazdaságtudományi Egyetem, Mechatronika, Optika és Gépészeti Informatika Tanszék, Budapest, Bertalan Lajos u. 4-6 D. épület, 1111, (06 1) 463 2602, urbin@mogi.bme.hu, https://www.mogi.bme.hu/
[3]Robert Bosch Kft., Budapest, Gyömrői út 104, 1103., (06 1) 879 9889, szilard.szoke@hu.bosch.com, https://www.bosch.hu/

Abstract

T3Ster provides an opportunity for thermal examination and identification in field of automotive electronics. T3Ster measurements can be performed more economically and competitively with robotization, which requires a vision system. This paper presents the hardware selection and the artificial intelligence (AI) based software development of the vision system using fully convolutional networks (FCN) and machine learning (ML) algorithms.

Keywords: neural network, measurement automation, t3ster, machine learning, vision system

Kivonat

A T3Ster egy olyan mérőeszköz, amely lehetőséget biztosít az autóipari elektronika területén a termikus tranziens tesztelésre és identifikációra. A T3Ster méréseket robotizációval gazdaságosabban és versenyképesebben lehet elvégezni, ehhez pedig látórendszer szükséges. Ez a cikk bemutatja a látórendszer hardverválasztását és a mesterséges intelligencia alapú szoftverfejlesztést, amely teljesen konvolúciós neurális hálókat és más gépi tanuláson alapuló algoritmusokat is használ.

Kulcsszavak: neurális háló, mérésautomatizálás, T3Ster, gépi tanulás, látórendszer

1. Bevezetés

A 21. század számos trendje, így a korszerű teljesítményelektronikát használó eszközök térhódítása, az autóipar nagyütemű elektrifikációja, illetve a villamosenergia-fogyasztási igények gyors növekedése miatt előreláthatólag továbbra is különös hangsúly helyeződik a közvetlen kötésű réz kerámiahordozók (direct bonded copper, DBC) iparágára. Ezeknek az eszközöknek az említett tendenciák miatt, egyre nagyobb teljesítmény disszipációs sűrűséggel kell megbirkózniuk, amely jelentős hőterhelésnek teszi ki őket. Az erre a terhelésre való méretezés gyakran egy biztonságkritikus feladat, ugyanis olyan jellegű alkalmazásokba kerülnek beépítésre, mint a szervomotor vagy az automata váltó, ahol különösen veszélyes az üzem folyamán bekövetkező meghibásodás. Ez komoly technológiai kihívást jelent, mivel éppen a túl magas hőmérséklet a vezető oka ezen eszközök meghibásodásának. Ezért a T3Ster [1] berendezés segítségével elvégezhető termikus

tranziens vizsgálatuk és a termikus identifikációjuk rendkívül fontos. A Robert Bosch Kft. ThID csapatával részt vettünk egy olyan projektben, amelynek célja az volt, hogy ezt a mérést automatizáljuk és felgyorsítsuk.

Fejlesztettünk egy látórendszerre alapuló összetett elektronikai eszközt egy mérőrobot számára, amelyet LabVIEW környezetben vezéreltünk. Rendszerünk segíthet a robotnak a DBC-k mérőtérben való felismerésében és megtalálásában. Ezt követően a robot automatikusan elvégezheti a szükséges méréseket. Mind a hardver, mind a szoftver fejlesztéséért felelősek voltunk. Egy megfelelő szenzorral és optikával ellátott kamerát választottunk, és létrehoztunk egy szerverplatformot, amely több számítógépet is kiszolgálhat egy Raspberry Pi 4 Model B segítségével, amelyek egyszerre irányíthatják vagy figyelhetik a mérést. Autodesk Inventor Professional 2022-ben terveztük meg a mechanikai integrációt támogató eszközt, majd kinyomtattuk 3D nyomtatóval. A munkánk leghangsúlyosabb részében mesterséges intelligencia, teljesen konvolúciós neurális hálók és számítógépes látáson alapuló algoritmusok segítségével végeztünk szemantikus szegmentációt és kulcspont keresést a robot navigációjához. Elkészítettük a szükséges adatkészletet és ezt követően alkalmaztuk a teljesen konvolúciós neurális hálókat. A tanító és validáló adatkészletet a valós méréshez kiválasztott kamera és optika segítségével állítottuk elő. Nagy hangsúlyt kapott a munka során a megfelelő neurális háló kiválasztása. Az OpenCV [2], Pytorch [3] és Torchvision [4] könyvtárakat használva, nyílt forráskódú Python 3 programozási nyelven, az Optuna könyvtár Fa-struktúrájú Parzen Becslője (Tree-structured Parzen Estimator, TPE) [5] szerinti automatizált futásokkal tanítottuk és értékeltük ki a különböző háló architektúrákat, módosítva az egyes hiperparaméter értékeket.

A cikk a következőkben a leírt projekt képfeldolgozási részére fókuszál, amely két részből áll. Az első rész a lencse és a szenzor kiválasztását részletező 2. szakasz. A második rész a szoftverfejlesztés, amely a 3. szakaszban kerül bemutatásra. Végül a 4. szakasz összefoglalja az eredményeket.

2. Kamerarendszer kiválasztása

Ebben a szakaszban a robot 250 µm pontosságú navigációját elősegítő rendszer optikájának és szenzorának kiválasztása kerül részletezésre.

2.1. A lencse kiválasztásához szükséges fókusztávolságtartomány kiszámítása

A 2. szakaszban említett pontossági követelmény két lépésben teljesíthető. Az első lépésben az optimális fókusztávolság tartományát kell megbecsülni. Ez azért becslés, mert az első lépésben a tárgytávolság és a lencsétől mért tárgytávolság különbsége ismeretlen, ezért ezeket azonosnak vesszük, és a második lépésben pontosítjuk a számítást. Továbbá feltételezzük, hogy az objektum az optikai tengely mentén helyezkedik el minden mérésnél. Tehát a műszaki optika törvényei alapján a szükséges fókusztávolság (f) tartományt kiszámíthatjuk a megfigyelni kívánt legkisebb méret pixel számából (a), a pixel méretből (px), a tárgytávolságból (s) és a legkisebb megfigyelendő méretből (x).

$$f = \frac{1}{\frac{x}{a \cdot px \cdot s} - \frac{1}{s}} \tag{1}$$

2.2. A szenzorválasztás

Miután kiválasztottuk a lencsét, választanunk kell egy szenzort is. A lencse műszaki paramétereinek alapján az a értéke kifejezhető a px pixel méret függvényében, és az s értéke megkülönböztethető a lencsétől mért tárgytávolságtól (s_0). A (2) egyenletben d a lencse teljes hossza. Ez lényegében a szenzor és a lencse eleje közti távolság.

$$a = \frac{x}{px} \cdot \frac{1}{|s_0 - d + f|} \cdot \frac{1}{\frac{1}{f} + \frac{1}{|s_0 - d + f|}} \tag{2}$$

Javasolt olyan px értéket választani, hogy az a minden lehetséges f esetén az [5; 15] tartományon belül maradjon. Az a érték kiválasztása két okból fontos. Ha túl nagy, akkor nem lesz megfigyelhető a teljes DBC. Ha pedig túl kicsi, akkor a legkisebb megfigyelendő méret nem lesz jól észlelhető. Az alkalmazott d bevezetése gyakorlati szempontból érdekes, mivel egy mérés konfigurációjánál az s_0 lesz a mérhető paraméter, nem pedig az s. Így a robot függőleges tengely mentén elfoglalt pozíciója eszerint állítható be.

3. Szoftverfejlesztés

A hardverválasztás elvégzése után az alkalmazás szoftveres implementációja következhet, ami konfigurációból, szemantikus szegmentációból és kulcspont alapú elmozdulás számításból áll.

3.1. Konfiguráció

A konfiguráció során eliminálju k a radiális torzítást. Ez nemlineáris Levenberg-Marquardt algoritmus alapú optimalizációval tehető meg [6] alapján. A kamera mátrixa SVD felbontással kerül kiszámításra. Egy másik kalibráció az optikai rendszer helyzetének és orientációjának kalibrációja. Ez az OpenCV beépített funkcióival [2] végezhető el, négy ismert pont alapján, amelyek segítségével perspektív transzformáció alkalmazható a kalibrációs képen.

3.2. A szemantikus szegmentáció és hiperparméter optimalizációja

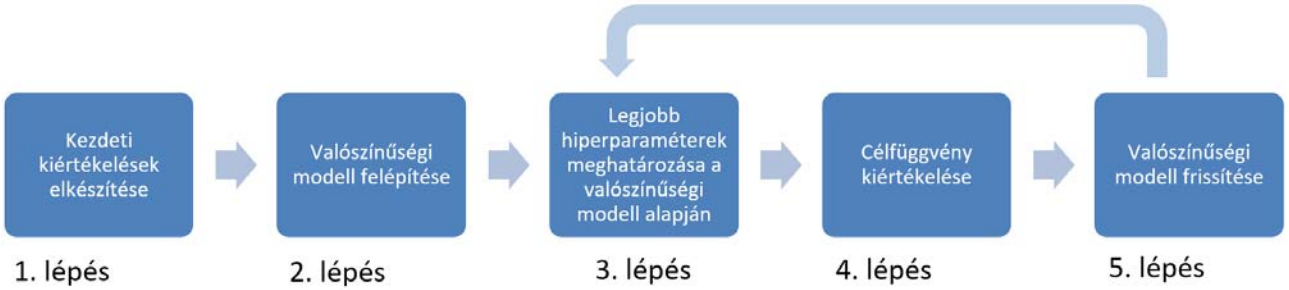
Miután a torzításmentes képet elkészítettük, szemantikus szegmentálást kell végezni annak meghatározásához, hogy hol található az a tartomány az adott képen (region of interest, RoI), ahol a kulcspontokat a navigációhoz meg kell találni. Egy részletes irodalmi kutatás eredményeként megállapítottuk, hogy az FCN-ek a legmegfelelőbbek erre a feladatra, mivel a hagyományos ML algoritmusok, mint például a K-means klaszterezés, nem adnak pontos megoldást [7]. A FCN-ek hatékonyságát alátámasztja [8], [9] és [10] is.

A szemantikus szegmentáció egy teljesen konvolúciós neurális háló segítségével történik. Ez leskálázásokból és felskálázásokból áll, amely konvolúciók és transzponált konvolúciók segítségével történik meg. Minden egyes konvolúciós réteg tartalmaz konvolúciót, batch normalizációt és ReLU [11] nemlinearitást. Az azonos méretű leskálázó konvolúciós rétegek és felskálázó transzponált konvolúciós rétegek között rövidzárak kerülnek alkalmazásra annak érdekében, hogy a költségfüggvénytől történő hiba-visszaterjesztés zavartalan lehessen. A 3x3-as konvolúciós rétegek a 2-es stride paraméternek köszönhetően mindig pont a felére skálázzák az adott aktivációs tömböt a szélesség és a magasság tekintetében, azonban a mélységet mindig kétszeresére növelik. Ennek megfelelően a háló hiperparamétereinek kiválasztása az 1. táblázat tartományaiból történik.

Optimalizálandó hiperparaméterek tartományai 1. táblázat

Hiperparaméter	Minimum érték	Maximum érték
Háló mélység	1	7
Csatornaszám	6	16
Tanulási ráta	10^{-4}	10^{-2}
Tanulási ráta csökkenési aránya	0,1	0,5
Tanulási ráta lépésköze	10	30
Regularizációs súlyozás	10^{-5}	10^{-3}
Batch méret	1	16
Dilatáció	0	1
Epoch szám	40	90

A háló mélység a leskálázások számát jelenti, míg a csatornaszám az első konvolúciós réteget követő aktivációs tömb mélységet. Ez ahogy az korábban is említésre került, minden leskálázás után duplázódik. Az optimalizáció az Adam [12] algoritmus segítségével történik, így a tanulási rátához, illetve regularizációhoz tartozó paraméterek ehhez kapcsolódnak. Dilatáció alkalmazása esetén a megfelelő mérettartás padding segítségével biztosított. A Fa-struktúrájú Parzen Becslővel történő hiperparaméter optimalizálás alapvetően a következő folyamatábrával írható le.



1. ábra.
A hiperparaméter optimalizáció folyamata

Látható, hogy az 1. és a 2. lépést csak egyszer kell elvégezni, míg a 3., 4. és 5. lépést ciklikusan ismételni kell. A megállási kritériumot jellemzően időhöz, iterációszámhoz, vagy célfüggvény értékhez szokás kötni. Az 1. lépés a szimulációs tér kezdeti véletlenszerű feltérképezését jelenti, hogy legyen alapja a valószínűségi modell felépítésének. Jelen munkában az Optuna [5] kísérletezési alapon meghatározott legjobb alapbeállítását, a 10 random iterációs beállítást használtuk. A 2. és 5. lépés a TPE specifikus modellezés, a 3. lépés a TPE specifikus kritérium alapján való döntés, a 4. lépés pedig a neurális hálós futtatás.

Érdemes elsőként a kritériummal foglalkozni. Mitől lesz jó egy hiperparaméter konstrukció? Ennek meghatározására a TPE a várható fejlődés (Expected Improvement, EI) függvényét használja:

$$EI_{y^*}(x) = \int_{-\infty}^{y^*} (y^* - \mathrm{y}) \cdot \mathrm{p}(\mathrm{y}|\mathrm{x})\mathrm{dy}, \tag{3}$$

ahol x az aktuális hiperparaméter konstrukció y^* a célfüggvény küszöbértéke, y a célfüggvény aktuális értéke $p(x|y)$ pedig a célfüggvény feltételes valószínűségi modellje. Már ebből a formulából is világosan látszik, hogy olyan célfüggvény értékeket lehet keresni ezzel az algoritmussal, amik kisebbek, mint a küszöbérték, hiszen ekkor lehet pozitív az integrál értéke. Tehát a hiba minimalizálandó. Ismerve a várható fejlődés fogalmát célszerű rátérni a TPE legfőbb elemére, a valószínűségi modellre:

$$EI_{y^*}(x) = \int_{-\infty}^{y^*} (y^* - \mathrm{y}) \cdot \frac{\mathrm{p}(x|y) \cdot \mathrm{p}(\mathrm{y})}{\mathrm{p}(\mathrm{x})}\mathrm{dy}, \tag{4}$$

ahol az eddig nem említett $p(x|y)$ a hiperparaméter konstrukció feltételes valószínűségi modellje, $p(y)$ a célfüggvény valószínűségi modellje, $p(x)$ pedig a hiperparaméter konstrukció valószínűségi modellje. Továbbá a TPE bevezeti $l(x)$ és $g(x)$ függvényeket úgy, hogy:

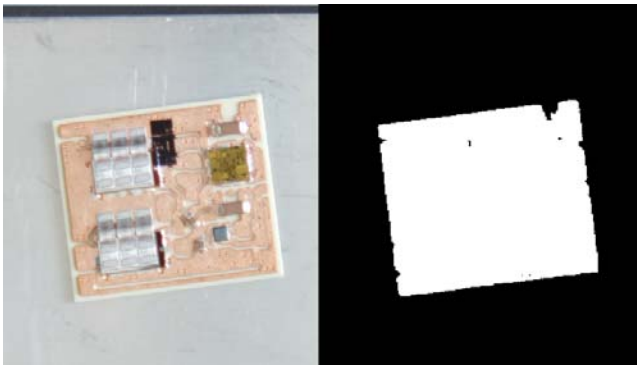
$$\mathrm{p}(x|y) = \begin{cases} l(x) \mid y < y^* \\ g(x) \mid y \geq y^* \end{cases} \tag{5}$$

Itt az $l(x)$ és $g(x)$ modellek gaussi keverékmodellként állnak elő úgy, hogy minden egyes adatra egy eloszlást kell illeszteni a hozzá jobbról és balról legközelebbi adatok közül a távolabb lévő távolságával megegyező szórással. Így ekvivalens átalakításokkal megkapható:

$$EI_{y^*}(x) = \frac{\mathrm{p}(\mathrm{y} < \mathrm{y}^*) \cdot \mathrm{y}^* - \int_{-\infty}^{y^*} \mathrm{p}(\mathrm{y})\mathrm{dy}}{\mathrm{p}(\mathrm{y} < \mathrm{y}^*) + \left(1 - \mathrm{p}(\mathrm{y} < \mathrm{y}^*)\right) \cdot \frac{\mathrm{g}(\mathrm{x})}{\mathrm{l}(\mathrm{x})}}. \tag{6}$$

Ebből pedig látszik, hogy ha a $g(x)/l(x)$ kifejezés csökken, akkor a (6) szerinti tört nevezője is csökken, ezáltal a tört, azaz a várható fejlődés nő, vagyis ekkor van a legnagyobb esély javítani a becslésen. Ez gyakorlatilag azt jelenti, hogy a küszöbérték feletti célfüggvény értékek küszöbérték alatti célfüggvény értékekhez viszonyított aránya csökken. Tehát gyakorlatilag a kisebb hibájú becsléseket eredményező hiperparaméter konstrukciókhoz közeli konstrukciót igyekszik választani mindig a TPE algoritmus az optimalizálásnál.

Az ezen módszerrel épített optimális neurális háló, 98,37%-os IoU értéket produkál, 2-es batch méret, 11-es csatornaszám, 5-ös mélység, 26-os lépésköz, 80-as epoch szám, 0,000583-as tanulási ráta, 0,000987-es regularizációs súlyozás és 0,37-es csökkenési arány mellett, dilatáció nélkül.



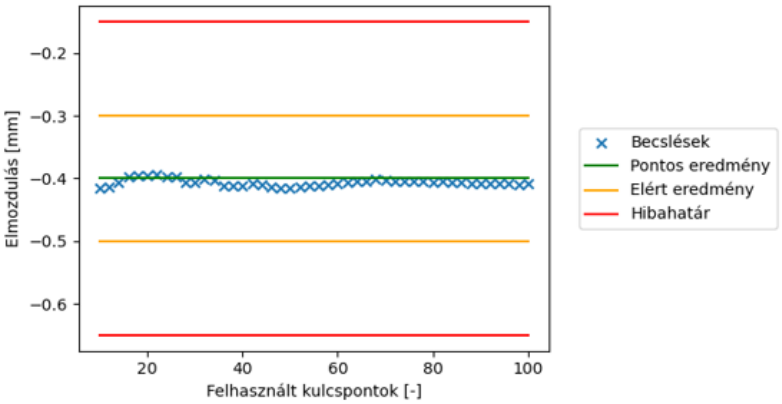
2. ábra.
A szemantikus szegmentáció eredménye

3.3. Kulcspontkeresés és elmozdulás számítás

Az RoI ismeretében a kulcspontok keresése és az elmozdulás meghatározása lehetséges azok koordinátái alapján. Ezeket a kulcspontokat a SIFT (Scale Invariant Feature Transform) detektor segítségével találtuk meg, amely elérhető az OpenCV-ben és a szabadalmi oltalma 2020 márciusában lejárt, így az iparban díjmentesen használható. Ez az algoritmus alapvetően sarokpontokat keres és invariáns leírókat hoz létre az összes transzformációra vonatkozóan a perspektív torzítást kivéve. A perspektív torzítás a kalibráció során kiküszöbölődik, így ez nem probléma. Mivel ezeknek a detektálásoknak a leírói jellemzik a létezésük valószínűségét is, ez a megközelítés képes korrigálni a kis szemantikus szegmentálásból eredő hibák hatását a munkafolyamatban a detektálások közül a legjobbak használatával.

4. Teljesítmény és áttekintés

Teljes rendszer felé fontos elvárás volt a biztos, robosztus működés. Éppen ezért részletes elemzést végeztünk, valós mérési körülményeket létrehozva. Az egyes DBC-k elmozdulása úgy került ellenőrzésre, hogy 10 µm pontosságú tolómérővel megmértük az egyes DBC pontok elmozdulását és ezek vektoraiból számítottunk elforgást is. Minden egyes DBC befoglaló téglalapjának csúcsait tekintve 4 pontot határoztunk meg, amelyeket kulcspontnak tekintve hajtottuk végre tolómérővel a mérés algoritmusát. Ezt 50 esetben végeztük el, ahol minden alkalommal megfelelő, nem csupán 250, hanem 100 µm-nél pontosabb becslést kaptunk. Ezek közül egy eset az alábbiakban kerül szemléltetésre. A függőleges tengely mentén az adott becsült és valós értékek összevetése látható, míg a vízszintes tengely mentén a 3.3 szakasz szerint felhasznált kulcspontok száma.



3. ábra.
Az elmozdulás becslésének eredménye

DBC-re ültetett alkatrészek esetében a szemantikus szegmentáció témája igen alacsony mértékben publikált, korszerűnek mondható, tudományos vizsgálatra érdemes. Kifejezetten DBC szemantikus szegmentációról nem is találtunk folyóirat-, vagy konferenciacikket. Hibakeresési módszer található a gyakran DBC-re helyezett IGBT-k körében [13], vagy nyomtatott áramkörök esetén is [14], azonban ez más technikát igényel. Más szakirodalom egyedi példányokkal foglalkozó (instance) szegmentációt végzett kapacitások, és ellenállások esetén [15]. Azonban ennek sikere ezen alkalmazás szempontjából kétes értékű lenne, mivel a közölt tanító és tesztelő adathalmazon egyszínű, fehér háttér előtt volt minden egyes komponens és az árnyékuk lehetett legfeljebb zavaró. Így nem lenne kellően robosztus a módszer alkalmazáshoz. Nyomtatott áramkörök esetén találhatók még érdekes, komponensekkel foglalkozó publikációk, azonban ezek nem szegmentációval, hanem detekcióval [16], vagy nagyobb elemek mélység alapú szegmentációjával foglalkoznak [17]. A fellelt elektronikához kapcsolódó cikkek közül egyik sem származik 2020-nál régebbről, így magabiztosan kijelenthető, hogy újszerű és nyitott témáról lehet beszélni.

Szakirodalmi vizsgálataink során tehát nem találtunk olyan munkát, amely ilyen nagy pontossággal valósított volna meg szemantikus szegmentációt és navigációt közvetlen kötésű réz kerámia hordozók esetén. Így munkánk alkalmazható a jövőben új ipari projektekhez, fejlesztésekhez. Továbbá újszerű megoldásnak számít a Fa-struktúrájú Parzen Becslő, a teljesen konvolúciós szemantikus szegmentációt végző neurális háló és a SIFT algoritmus együttes használata, melyekkel a felsorolt eredményeket elértük.

Irodalmi hivatkozások

[1] ***: *Characterize the thermal properties of components*. Simcenter Micred T3STER, https://plm.sw.siemens.com/en-US/simcenter/physical-testing/t3ster/ (Utolsó letöltés: 2023. 02. 27).

[2] Bradski, G. *The OpenCV Library*. Dr. Dobb's Journal of Software Tools. 2000.

[3] Paszke, A. et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Advances in Neural Information Processing Systems 32. Curran Associates, Inc., 2019, 8024--8035.

[4] Fabel D.: *torchvision: Models, Datasets and Transformations for Images*. Torchvision, https://torchvision.mlverse.org/ (Utolsó letöltés: 2023. 02. 27).

[5] Akiba T., Sano S., Yanase T., Ohta T., Koyama M. *Optuna: A Next-generation Hyperparameter Optimization Framework*, Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.

[6] Drap P, Lefèvre J. *An Exact Formula for Calculating Inverse Radial Lens Distortions*. Sensors. 2016, 16(6), 807-824.

[7] Békési G. *MOSFET-ek és más elektronikai komponensek T3Ster mérésének automatizálása mesterséges intelligencia, neurális hálók és számítógépes látás segítségével*. Tudományos Diákköri Konferencia. BME, 2021.

[8] Guo Y., Liu Y., Georgiou T., Lew M. S. *A review of semantic segmentation using deep neural networks*, International Journal of Multimedia Information Retrieval, 2018. 7, 87-93.

[9] Lateef F., Ruichek Y., *Survey on semantic segmentation using deep learning techniques*, Neurocomputing, 2019. 338, 321-348.

[10] Li B., Shi Y., Qi Z., Chen Z. *A Survey on Semantic Segmentation*, IEEE International Conference on Data Mining Workshops (ICDMW), 2018. 1233-1240.

[11] Agarap, A. F. *Deep learning using rectified linear units (relu)*, 2018. arXiv preprint arXiv:1803.08375.

[12] Kingma D. P., Ba J. *Adam: A method for stochastic optimization*, ICLR, 2015.

[13] Li Y., Liu S., Li C., Zheng Y., Wei C., Liu B., Yang Y. *Automated defect detection of insulated gate bipolar transistor based on computed laminography imaging*, Microelectronics Reliability, 2020. 115.

[14] Shi W., Lu Z., Wu W., Liu H. *Single-shot detector with enriched semantics for PCB tiny defect detection*, The Journal of Engineering, 2020. 13, 366-372.

[15] Yang Z., Dong R., Xu H., Gu J. *Instance Segmentation Method Based on Improved Mask R-CNN for the Stacked Electronic Components*, Electronics, 2020. 9(6), 886-901.

[16] Lian J., Wang L., Liu T., Ding X., Yu Z. *Automatic visual inspection for printed circuit board via novel Mask R-CNN in smart city applications*, Sustainable Energy Technologies and Assessments, 2021. 44.

[17] Li D., Li C., Chen C., Zhao Z. *Semantic Segmentation of a Printed Circuit Board for Component Recognition Based on Depth Images*, Sensors, 2020. 20, 5318-5334.