Gépi tanulási módszerek optimalizálása a stencilnyomtatási folyamat vizsgálatára

Martinek Péter, Krammer Olivér

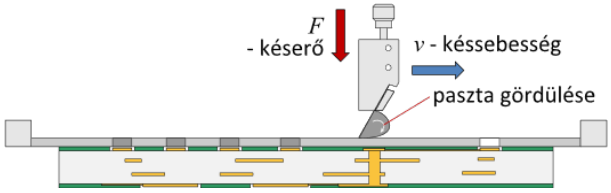
Budapesti Műszaki és Gazdaságtudományi Egyetem
Elektronikai Technológia Tanszék, 1111 Budapest, Egry József u. 18.
E-mail: krammer@ett.bme.hu

***Tartalmi kivonat**. Jelen cikk célja, hogy bemutassa az újraömlesztéses forrasztási technológia kritikus lépésének, a stencilnyomtatási folyamatnak a gépi tanuláson alapuló modellezési lehetőségeit. Ismertetjük az egyes gépi tanulási módszerek alapjait, majd pedig a következő gépi tanulási módszerek optimalizálását a stencilnyomtatás tekintetében: mesterséges neurális hálózat, neuro-fuzzy rendszer, szupport-vektor gépek, boost-olt döntési fák. A módszerek vizsgálatához és optimalizáláshoz kísérleti úton nyertük a tanító adathalmazt, melynek bementi paraméterei a forraszpaszta szemcseméretének tulajdonságai, a stencilapertúra mérete és a nyomtatási sebesség. A folyamat minőségét jellemző kimeneti paraméterek pedig a forraszpaszta-lenyomatok területe, magassága, térfogata. Az egyes, gépi tanulási módszerek becslési hibáját az átlagos abszolút százalékos hiba (MAPE – mean absolute percentage error) értékével jellemeztük. A vizsgált gépi tanulási módszerek teljesítményét összességében megfelelőnek találtuk (az átlagos becslési hiba 5% alatti), kivéve a neuro-fuzzy rendszert, melynek alkalmazását nem javasoljuk a stencilnyomtatási folyamat modellezésére.*

***Kulcsszavak:** stencilnyomtatás; gépi tanulás; mesterséges neurális hálók; döntési fák; neuro-fuzzy rendszerek; szupport-vektor gépek*

1. Bevezetés

Napjainkban az elektronikai eszközök tömeggyártását az automatizált felületszerelési technológia uralja, melynél a nyomtatott áramköri lemezre az alkatrészeket ún. újraömlesztéses forrasztással rögzítik [1,2]. Ezen technológia lényege, hogy a forraszanyagot (2006 óta egyre elterjedtebben ólommenteset) forraszpaszta formájában viszik fel a szerelőlemez kontaktusfelületeire, ebbe ültetik bele az elektronikus alkatrészeket, majd végül a szerelvény áthalad egy alagútkemencén. A kemencében a forrasz olvadáspontja fölé melegítik a teljes szerelvényt, a forrasz mindenhol megolvad, majd lehűtés és megszilárdulás után létrejönnek a forrasztott kötések. A modern áramkörökben az alkatrészméretek folyamatosan csökkennek, hogy az áramköri paraméterek megfeleljenek a hordozható-, az IoT- (Internet of Things) és az 5G eszközök követelményeinek. Az alkatrészek csökkenő mérete (pl. a passzív alkatrészeknél 200 x 100 µm) komoly kihívást jelent a forrasztási technológia legkritikusabb lépésénél, a forraszpaszta (folyasztószer és forrasz-szemcsék szuszpenziója) felvitelére szolgáló stencilnyomtatás során [3,4]. A stencilnyomtatási folyamat során megadott sebességgel és megadott erővel húzzuk végig a nyomtatókést a stencil felületén. Ennek hatására a forraszpaszta mozgásba jön, gördül a kés előtt, és kitölti a stencilen lévő, a forrasztási felületek felett elhelyezkedő apertúrákat (1. ábra).



1. ábra. A stencilnyomtatás folyamatának sematikus ábrája

A stencil és a hordozó szétválasztása után a forraszpaszta a szerelőlemez kontaktusfelületeire kerül. Kutatások szerint a gyártási hibák akár 50–60%-a is erre a folyamatra vezethető vissza [5]. A nyomtatási folyamat még kritikusabbá vált az ultrafinom raszterosztású, mint pl. QFN (Quad-Flat-No-Lead) és µBGA (Micro Ball Grid Array) [6] tokozású alkatrészek széleskörű elterjedésével, mert ezen alkatrészekhez még kisebb méretű apertúrák tartoznak a stencilen, mint a passzív alkatrészekhez. Ezért elengedhetetlen a stencilnyomtatási-folyamat alapos, új módszerekkel történő vizsgálata pl. gépi tanulás alkalmazásával, az ún „zero-defect”, nulla-hibás gyártás eléréséhez.

2. A GÉPI TANULÁSI MÓDSZEREK

2.1. Gépi tanulási módszerek alapjai

A gépi tanulási módszerek a mesterséges intelligencia egy részhalmazát képező olyan algoritmusok, melyek tapasztalati úton képesek az ismert adatok alapján fejlődni és egyre pontosabb eredményeket nyújtani. Az úgynevezett tanuló adathalmaz alapján az algoritmusok olyan modellt hoznak létre, mely képes eredményeket kis hibával előre jelezni vagy éppen megfelelő döntést hozni anélkül, hogy közvetlenül az adott problémára, illetve annak megoldására fejlesztettük volna (programoztuk volna). A gépi tanulás tipikusan számítógépeket használ és magát a tanulást úgy valósítja meg, hogy a számítógép előre meghatározott lépések sorozatát hajtja végre, melyek elvezetnek a probléma megoldásához anélkül, hogy a számítógép „ténylegesen” tanulna. Három alapvető megközelítést különböztetünk meg a gépi tanulási módszerek között:

- felügyelt tanulás esetén egy megadott adathalmaz alapján tanul az algoritmus és előrejelzéseket készít további új bemeneti értékekhez tartozó kimenetre,
- felügyelet nélküli esetben a bemenet nem strukturált, nincs előre felcímkézve; így a feladat elsősorban az adatok struktúrájának kinyerése és az összefüggések megkeresése az adathalmazban,
- megerősítéses tanulásnál az algoritmusok valamely kimeneti értéket-értékeket próbálják optimalizálni (pl. minimalizálni vagy maximalizálni) és ezen érték visszajelzése alapján a problémateret bejárva választanak a következő lehetséges végrehajtandó lépések közül egy vélhetően számukra kedvezőt.

A gépi tanulás alkalmazásának egyik legfőbb követelménye, hogy minél nagyobb mennyiségű adat álljon rendelkezésre. Továbbá fontos az adatok gyors elérhetősége és változatossága és az adat általános minősége is. A gépi tanulás alkalmazásának a sikertelenségét a legtöbb esetben e szükséges feltételek (röviden: mennyiség, sebesség, változatosság) teljesítésének hiánya okozza.

A mennyiséggel kapcsolatban sokszor a legnagyobb problémát az okozza, hogy nem tudunk elegendő adatot teremteni. Az adat mennyiségének növekedésével ugyanakkor a sebesség is különösen fontos összetevővé válik, hiszen a becslés hatékonyságát nem csak a becsült érték minősége adja, hanem a becslés sebessége is meghatározó lehet. A gépi tanulási módszerek legfőbb előnyei az alábbiak:

- könnyen és gyorsan tudunk különböző trendeket és összefüggéseket találni akár nagy adathalmazokon is,
- nincs szükség futás közben emberi beavatkozásra, az algoritmus magától „tanul”,
- minél nagyobb adathalmazon, minél tovább „tanul” az algoritmus annál hatékonyabban működik és pontosabb eredményeket ad,
- egyszerűen kezeli a többdimenziós és többváltozós problémákat is és
- szinte bármilyen területen alkalmazható, vagy elérhető már az adott területre finomhangolt kész megoldás.

A legfőbb hátrányok között említhetjük a következőket:

- nagy mennyiségű, jó minőségű adatra van szükség a megfelelő működéshez, ami nem biztos, hogy rendelkezésre áll, vagy előállítható rövid időn belül, alacsony költségen,
- a jó eredményhez gyakran sok időre és sok számítási kapacitásra van szükség,
- az eredmények értelmezése nagy körültekintést igényel és fontos a megfelelő módszer és algoritmus kiválasztása és
- adott esetben rendkívül nagy lehet a módszerek hibája is, például nem megfelelő méretű tanítóhalmaz esetén.

Mivel az általunk vizsgált stencilnyomtatási folyamat nemlineáris, több bemeneti és kimeneti értékkel rendelkezik így a megfelelő gépi tanulási módszer, jó bemeneti adathalmazon vélhetően kis hibával tud előrejelzést készíteni a kimeneti értékekre a betanítás után. A kutatásunkban több gépi tanulási módszert vizsgáltunk meg részletesen a stencilnyomtatási folyamat optimalizálása céljából. A következő alfejezetekben részletesen bemutatjuk ezen megközelítések elméleti hátterét és az adott megközelítés finomhangolását a stencilnyomtatás folyamatára. Ez utóbbi magában

foglalja az optimális méretű és felépítésű struktúra azonosítását, valamint a tanítást befolyásoló paraméterek finomhangolását a teljesítmény – pontosság – növelése és lehetséges maximalizálása érdekében.

2.2 Mesterséges neurális hálózatok

Az egyik legnépszerűbb gépi tanulási technika a mesterséges neurális hálózat, amelyet széles körben alkalmaznak bármilyen típusú kimeneti paraméterek előrejelzésére. A modell mesterséges neuronokat tartalmaz, melyek felfogják a külvilágból érkező ingereket és jeleket küldenek a kapcsolódó csomópontokhoz. A modellezés során az elfogadott bemeneti jelek többnyire valós számok, melyek függvényében határozzuk meg a kimeneti értékeket melyek legtöbbször szintén konkrét számértékek. A struktúrát szigorú sorrendben határozzuk meg, ahol a neuronok csak a szomszédos rétegek között kommunikálnak egy adott irányban. Egy csomópont aktivációs függvényének bemenete formálisan a következőképpen határozható meg (1):

$$z_j^l = \sum_k \left(w_{jk}^l a_k^{l-1} \right) + b_j^l , \qquad (1)$$

ahol w_{jk}^l a k. neuron súlya az (l-1). rétegben az l. réteg j. neuronjához kapcsolódva, b_j^l a j. neuronhoz tartozó eltolási (bias) érték, a_k^{l-1} a k. neuron kimeneti értéke az (l-1). szinten és z_j^l a j. neuron aktivációs függvényének bemenete az l. szinten [7]. Az aktivációs függvény kimenete $a_j^l = \sigma\left(z_j^l\right)$, ahol $\sigma(z)$ a neuron aktivációs függvénye. A mesterséges neurális hálózat struktúrájának meghatározása (rétegek és neuronok száma, valamint ezek összerendelése) után a betanítás következik mely az adott feladat mintáit felhasználva hangolja be a megalkotott szerkezetet. Számos betanítási módszer létezik a szakirodalomban, melyeket implementálva is könnyen megtalálhatunk elterjedt szoftver eszközökben, mint például a *Neural Designer*, a *Neuroph* vagy éppen olyan robusztus eszközökben, mint a *Weka* vagy a *Matlab*.A mesterséges neurális hálózatok egyik elterjedt tanítási módszere a Bayes-i regularizáció [8], melynél a struktúra súlyai a Bayes szabály alapján kerülnek meghatározásra az egyes iterációkban (2).

$$P(\mathbf{w} \mid D, \alpha, \beta, M) = \frac{P(D \mid \mathbf{w}, \beta, M) P(\mathbf{w} \mid \alpha, M)}{P(D \mid \alpha, \beta, M)}, \qquad (2)$$

ahol D a betanítási adathalmaz, M az adott neurális hálózat modellje és $\mathbf{w}$ az élsúlyokat tartalmazó vektor. Az előzetes sűrűségfüggvény $P(\mathbf{w} \mid \alpha, M)$ tartalmazza az induló súlyértékeket és $P(D \mid \mathbf{w}, \beta, M)$ maga a likelihood függvény, ami az adat előfordulási valószínűségét adja adott $\mathbf{w}$ súlyértékek mellett. $P(D \mid \alpha, \beta, M)$ a normalizációs faktor, ami biztosítja, hogy az összvalószínűség pontosan 1 legyen [9]. Mind a tanítóhalmazban fellelhető zajt mind pedig a súlyok előzetes eloszlását Gauss eloszlásúnak feltételezzük. Így a Bayes szabályt alkalmazva optimalizáljuk a célfüggvény α és β paramétereit az alábbi kifejezést alkalmazva (3).

$$P(\alpha, \beta \mid D, M) = \frac{P(D \mid \alpha, \beta, M) P(\alpha, \beta \mid M)}{P(D \mid M)} \qquad (3)$$

Habár a Bayes-i regularizációs módszer nagyon jó eredményeket adhat még kevés és zajos bemeneti értékek mellett is, maga a tanítás nagyon sok időt is igénybe vehet. Ez utóbbi probléma kiküszöbölésére lehet jó megközelítés a Levenberg–Marquardt tanítási módszer, ami nagyságrendekkel kevesebb futásidő mellet is adhat jó eredményt [10]. A Levenberg–Marquardt módszer a legkisebb négyzetek illesztési megközelítésén alapul a következők szerint. Jelölje (x_i,y_i) a bemeneti és kimeneti értékpárokat és próbáljuk megtalálni a $\boldsymbol{\beta}$ paramétert az $f(\mathbf{x}, \boldsymbol{\beta})$ görbéhez az $S(\boldsymbol{\beta})$ eltérés minimalizálásával (4):

$$\hat{\beta} = \arg\min_\beta S(\boldsymbol{\beta}) = \arg\min_\beta \sum_{i=1}^{m} \left[y_i - f(x_i, \boldsymbol{\beta}) \right]^2 . \qquad (4)$$

Az iteratív megoldási módszerben a $\boldsymbol{\beta}$ paramétervektor egy új $\boldsymbol{\beta} + \boldsymbol{\delta}$ becslésre lesz cserélve minden lépésben. A $\boldsymbol{\delta}$ meghatározásához pedig a $f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta})$ függvényt annak linearizációjával közelítjük az alábbi módon (5):

$$f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta}) \approx f(x_i, \boldsymbol{\beta}) + J_i \boldsymbol{\delta} . \qquad (5)$$

ahol $J_i = \dfrac{\partial f(x_i, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ f gradiens vektora $\boldsymbol{\beta}$ függvényében. A Levenberg-Marquardt algoritmus képes jó eredményt adni a Bayes-i regularizációhoz képest sokkal rövidebb futásidő alatt. Cserébe elképzelhető, hogy sokkal több működési memóriára van szüksége a tanítás alatt, ami jelenthet

szűk keresztmetszetet nagy elemszámú tanítóhalmazok esetén. A szakirodalomban és az elérhető szoftver eszközökben sok más képzési módszer is található, mint például a BFGS Quasi-Newton, a Scaled Conjugate Gradient [11], a Conjugate Gradient with Powell/Beale Restarts, a Fletcher-Powell Conjugate Gradient vagy a Polak-Ribiére Conjugate Gradient módszer. Mivel korábbi kutatásaink során a Levenberg-Marquardt és a Bayes-i regularizáció módszerek egyértelműen jobb eredményeket adtak más módszereknél, így jelen kutatásban is csak ezt a két módszert vizsgáltuk részletesen, így a többi részletes bemutatásától ezen szakaszban most eltekintünk.

2.3. Döntési fák

A döntési fák felépítik a tanítási halmaz példányainak kategorizálását azáltal, hogy jól definiált igaz/hamis kérdésekből képeznek elágazásokat fa struktúra formájában. A döntési fa struktúrájában a levelek a megfelelő példánykategóriát képviselik, az ágak pedig a jellemzők összekapcsolódását jelentik, amelyek ezekhez a kategóriákhoz vezetnek. Ha a célérték folytonos, pl. valós számokat tartalmaz, akkor a fát regressziós fának nevezzük. Az elterjedt vonatkozó kifejezés, a CART (Classification and Regression Trees) mindkettőre kiterjed [12]. A döntési fák legfontosabb előnye, hogy az adatok nem igényelnek semmilyen előkészítést, és a módszer könnyen alkalmazható nagy adathalmazon is. A módszer pontossága és robusztussága azonban alacsonyabb lehet más megközelítésekhez képest. Az ún. hibrid modellekben alkalmazott technikák (pl. boosting vagy bagging) segítenek leküzdeni ezeket a hátrányokat, míg a gradiens növelés segít a szükséges alacsony számítási idő fenntartásában.

A gradiens boostolás ötlete iteratív módszeren alapszik, amelynek képzési bemenete $\{(x_i, y_i)\}_{i=1}^{n}$, a veszteségfüggvénye pedig $L(y, F(x))$. A modell $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$ finomítása lépésenként történik, ahol a tanuló egységet, például a döntési fát – $h_m(x)$ igazítjuk a következő optimalizációs problémát megoldva (6):

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(\mathbf{x}_i) + \gamma h_m(x_i)), \qquad (6)$$

ahol $F_m(x)$ a módosított modell az *m*. iteráció után. A gradiens boostolás során rögzített méretű fákat használunk, ahol az alaptanuló (base learner) egy $h(\mathbf{x})$ regressziós fa melynek *L* levele van. Ez a regressziós fa az **x** vektorteret *L* független régióra $\{R_{lm}\}_{l=1}^{L}$ osztja minden *m*. iteráció során. A minimalizációs probléma megoldása mindezek alapján [13] (7):

$$\gamma_{lm} = \arg\min_{\gamma} \sum_{\mathbf{x}_i \in R_{lm}} L(y_i, F_{m-1}(\mathbf{x}_i) + \gamma). \qquad (7)$$

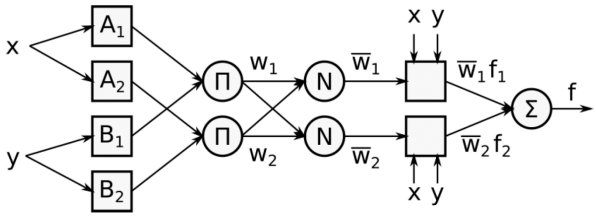
Elképzelhető egyszerű regularizációs eljárások alkalmazása is, mely az alaptanulók hozzájárulását határozza meg *v* faktor segítségével [13] a következő módon (8):

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + v \cdot \gamma_{lm} I(\mathbf{x} \in R_{lm}), \qquad (8)$$

ahol *v* paramétert gyakran úgy emlegetik, mint a "tanulási ráta" (learning rate).

2.4. Neuro-fuzzy rendszerek

A neuro-fuzzy rendszerek a neurális hálózatokat és a fuzzy döntési logikát kombinálja nemlineáris függvények közelítése céljából. A struktúra számos fuzzy operátort tartalmaz melyek adott számú tagsági függvényt kapcsolnak össze. Az így létrejött hálózatban különböző "élsúlyokat" és egyéb paramétereket is definiálunk, melynek eredményeként az ANFIS (adaptív neuro-fuzzy rendszer) egy robusztus és pontos becslési modell lehet. Az ANFIS rendszer elméleti blokkdiagrammját a 2. ábra mutatja.



2. ábra. Az ANFIS elméleti blokkdiagrammja [14]

A 2. ábrán mutatott példa rendszerben az alábbi szabály létezik a struktúrára vonatkozóan:

Ha *x* értéke A_1 és *y* értéke B_1 akkor (9):

$$f_1 = p_1 x + q_1 y + r_1, \qquad (9)$$

ahol *x* és *y* jelöli a bemeneteket, f_1 a kimenetet, A_1 és B_1 a két kapcsolódó tagsági függvény és p_1, q_1 valamint r_1 a kapcsolódó paraméterek. Az első réteg 2 differenciálható tagsági függvény csomópontot (A_i

$\overline{w_1}$

súlyokkal. A 2. ábrán látott példa szerint a súlyokat így számíthatjuk ki: $\overline{w_1} = \frac{w_1}{w_1 + w_2}$ valamint $\overline{w_2} = \frac{w_2}{w_1 + w_2}$. A 3. rétegben így kapott értékek a fenti szabály szerint lesznek f_1 értékeivel megszorozva. Az utolsó réteg a kimenet előállításáért felelős a bejövő jelek összegzésével, mely réteget a 2. ábrán ∑-val jelöltünk. Ezt a réteget szokás kimeneti rétegnek is nevezni.

A mesterséges neurális hálózattól eltérően, itt az egyes tagsági függvényeket a különböző bemenetekhez is kell rendelnünk, azaz a számuk meghatározása után fel is kell osztanunk őket a bemenetek szerinti halmazokba. Ezt követi aztán a neurális hálózatokhoz hasonlóan a tanítás folyamata, ahol az ANFIS működési paramétereit is be kell állítani az adott tanulási minták alapján. A tagsági függvények paramétereinek meghatározása tehát a tanítás során történik majd.

2.5. Szupport-vektor gépek

A szupport-vektor gép (SVM) egy felügyelt tanulási algoritmus, amelyet számos osztályozási és regressziós probléma esetén használnak, legyen szó akár a jelfeldolgozó alkalmazásokról, vagy éppen természetes nyelvek feldolgozásáról, illetve kép- és beszédfelismerésről. Az SVM algoritmus célja olyan hipersík megtalálása, amely a lehető legnagyobb mértékben elválasztja az egyik osztály adatpontjait a másik osztálytól. Az optimális az a hipersík, amely a legnagyobb különbséggel rendelkezik a két osztály között. A támogató vektorok a tanítás egy részhalmazára hivatkoznak, amelyek meghatározzák az elválasztó hipersík helyét. A standard SVM algoritmus bináris osztályozási problémákra lett megfogalmazva, így a többosztályos problémák jellemzően bináris sorokká redukálódnak.

A szupport-vektor gépek a gépi tanulási algoritmusok úgynevezett kernelmódszerek csoportjába tartoznak, ahol a funkciókat kernelfüggvény segítségével lehet átalakítani. A kernel-függvények az adatokat egy másik, gyakran magasabb dimenziós térbe képezik le, azért, hogy az osztályok könnyebben elkülöníthetőek legyenek ezen átalakítás után. A komplex, nemlineáris döntési határokat tehát például lineáris, magasabb dimenziós leírásokra cserélhetjük. Ez közismert nevén kernel trükk (kernel trick). Többféle kernel is létezik az SVM-ek esetén, például lineáris kernel két tanuló osztállyal (10), polinomiális kernel (11), Gauss-i kernel (12).

$$K(x_1, x_2) = x_1^T x_2 . \tag{10}$$

$$K(x_1, x_2) = \left(x_1^T x_2 + 1\right)^{\rho} . \tag{11}$$

$$K(x_1, x_2) = \exp\left(-\frac{\left\|x_1 - x_2\right\|^2}{2\sigma^2}\right)^{\rho} , \tag{12}$$

ahol ρ a polinom foka.

A szupport-vektor gép betanítása a lenti négyzetes optimalizálási probléma (13) megoldását jelenti. Keressük azt az **x** vektort, ami minimalizálja a következő kifejezést:

$$\min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T H \mathbf{x} + f^T \mathbf{x} \right\} \tag{13}$$

a következő kényszerek mellett: $Ax \le b$, egyenlőtlenségi kényszer; $A_{eq} x = b_{eq}$, egyenlőségi kényszer; és $lb \le x \le ub$, (alsó-felső) korlát kényszer. A szupport vektor gépek legfőbb előnyei, hogy magasabb dimenziószámok esetén is alkalmazhatóak és gyors becslést tesznek lehetővé mind osztályozási mind regressziós feladatok esetén. Ugyanakkor mivel kevésbé skálázható, így leginkább csak kisebb adathalmazok esetén használható, és fennáll továbbá a túltanítás (overfitting) esélye. Mindemellett a struktúra betanítása nemlineáris kernel esetén eléggé számításigényes is lehet.

3. GÉPI TANULÁS – TANÍTÁSI MÓDSZER

A tanításhoz a korábban megjelentetett cikkünkben [15] ismertetett kísérletből nyertük az adathalmazt, amely 15 675 bemeneti értékkombinációhoz rendeli hozzá a mért kimeneti értékeket. A bemeneti adatok dimenziói rendre a forraszpaszta típusát jellemző átlagos szemcseméret és kapcsolódó szórás érték, a nyomtatási sebesség, valamint a stencilapertúrák terület-falfelület aránya az úgynevezett AR (area ratio). A gépi tanulási módszertől függően ezt a halmazt a tanítás során

tanítási, teszt és validációs halmazokra is bontottuk. Módszertől függően elképzelhető, hogy ez a véletlenszerű felosztás többször és különbözőképpen is megtörténik a tanítás egyes fázisaiban – iterációiban. Ugyanakkor hangsúlyozva a gépi tanulási módszerek használatának egyik nagy előnyét megjegyezzük, hogy semmilyen más előzetes formázási, átalakítási feladat nem merül fel az adatokkal kapcsolatban.

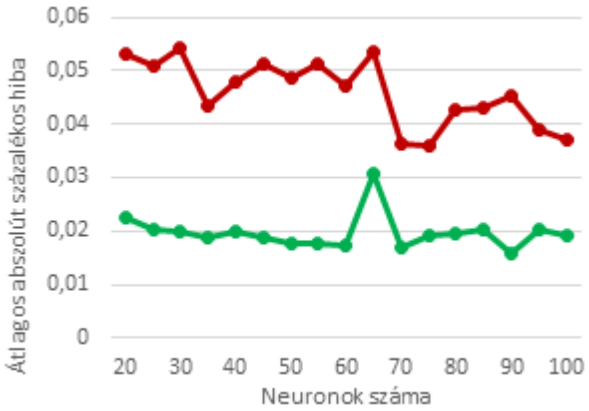
Az adathalmaz kimeneti értékei a felvitt forraszpaszta területe, magassága és térfogata. Azon módszerek esetén, ahol csak egy kimeneti paramétert képes becsülni a modell ott több modell készítésével és a kimeneti dimenziók egyenkénti becslésével oldottuk meg a feladatot – a szakirodalomban ilyenkor alkalmazott megközelítéssel egybehangzóan. A kísérleteknél ilyenkor egyértelműen megjelöljük, hogy a tanítás mért futásideje egy kimeneti változóhoz tartozó modell tanítására vagy mindhárom kimeneti értékhez tartozó modell tanítására vonatkozik majd. A tanítási folyamat során a becslési hiba minimalizálását végezzük el. A legtöbbször használt hibamutató itt az átlagos négyzetgyök hiba (RMSE – root mean squared error) volt.

A végeredmény validációjához ugyanakkor készítettünk egy 345 elemű bemeneti-kimeneti adathalmazt, mely szigorúan különböző bemeneti kombinációkat tartalmaz, melyre a kimenetet a kísérletek adott bemeneti értékeihez tartozó kimenetek számtani közepeként határoztuk meg. A validációs adathalmaz előállításakor a hasonló AR értékekhez tartozó adatsorokat egy kerekített középértékhez rendelve össze is vontuk. Ebben az ellenőrzésben a gépi tanulási módszerek hibáját az átlagos abszolút százalékos hibamutatóval (MAPE – mean average percentage error) fejeztük ki.

4. Eredmények

4.1. Mesterséges neurális hálózat vizsgálata

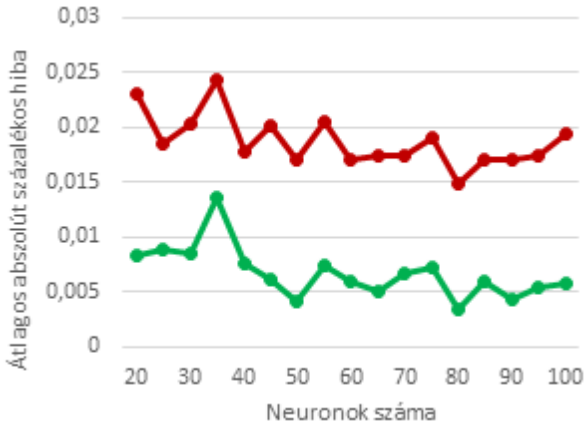
A mesterséges neurális hálózat adott problémára optimális struktúra méretét és a leghatékonyabb tanítási módszert kísérleti úton határoztuk meg. Mivel egyszerű regressziós probléma becsléséről van szó, így egy rejtett neuron réteg az előrejelzéshez megfelelő. A struktúra méreténél kérdés csupán ezen réteg által tartalmazott neuronok száma. Egyszerű megközelítések alapján a lehetséges neuronok számát 10 és 200 közötti értékeknél vizsgáltuk az előrejelzési pontosság meghatározásával. A tanító algoritmusok közül a Bayes-i regularizáció (BR) és a Levenberg–Marquardt (LM) módszerek hatékonyságát vizsgáltuk részletesebben. Míg az előző pontosabb eredményeket adott kis neuronszámok esetén is, addig utóbbi a több neuronból álló struktúrák esetén is képes volt eredményt adni elfogadható futásidő mellett. Az LM módszer pontosságát a neuronok számának függvényében a 3. ábrán mutatjuk. Mivel az egyes tanításokra jelentősen különböző pontosságú eredményeket is kaphatunk, így minden neuronszám mellett 5 tanítást végeztünk és ezek eredményét átlagoltuk. Az eredményekhez kapcsolódó szórás értékét is feltűntettük a 3. ábrán.



3. ábra. MAPE érték alakulása a neuronszám függvényében LM képzésnél (piros: hibaérték; zöld: szórás)

A módszer által elért legjobb pontosság 75 neuronszámnál figyelhető meg, melynek átlagos értéke 3,6%. Ebben a magasság becslési hibája némileg alacsonyabb, 2,2%-os, míg a térfogat becslése némileg rosszabb 4,8%-os hibát (MAPE érték) mutatott. A terület becslési pontossága 3,8% volt. Kísérleteink szerint ennél lényegesen jobb pontosságot az LM tanítási módszerrel nagyobb neuronszámok mellett sem érhetünk el: a vizsgált 100-200 neuronszámtartományban az elért legjobb átlagos hiba 3,3% volt, ugyanakkor a futásidő akár több óra is lehetett, így ezen tartomány eredményeit részletesen most nem mutatjuk be.

A Bayes-i regularizáció tanító módszerénél már kisebb hibájú becsléseket is kaptunk. Az eredményeket a 4. ábrán ismertetjük. Mivel a módszer futásideje meglehetősen nagy ugyanakkor az egyes neuronszámokra betanított hálózatok esetén nem mutatkozik nagy eltérés a hibában, így neuronszámonként csak 1-1 tanítást végeztünk.



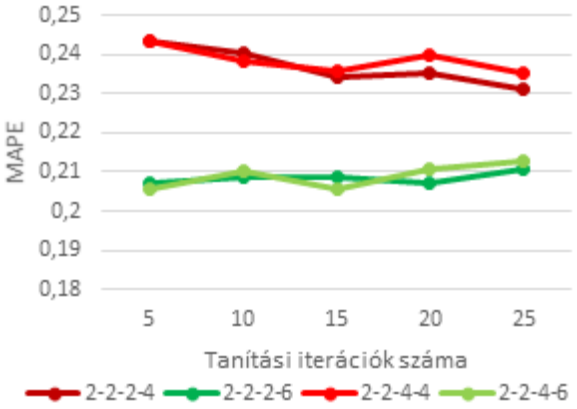
4. ábra. MAPE érték alakulása a neuronszám függvényében BR képzésnél (piros: hibaérték; zöld: szórás)

A módszer által elért legjobb pontosság 80 neuronszámnál figyelhető meg, melynek átlagos MAPE értéke 1,5%. Ez az érték alacsonyabb, mint az LM tanításánál, cserébe a tanítási idő hosszabb, 5–10 perces nagyságrendben mozog. Az átlagos hibaértéken belül a terület és a magasság becslési hibája némileg alacsonyabb 1,3% ill. 1,2%, míg a térfogat becslése némileg rosszabb, 1,9%-os hibát mutatott. A kísérletek alapján megállapítható, hogy a neurális hálózatok segítségével képesek vagyunk a stencilnyomtatási folyamat megfelelő modellezésére és a kimeneti paraméterek becslésére alacsony (átlagosan 1,5%) hiba mellett.

4.2. Neuro-fuzzy rendszer optimalizálása

A neuro-fuzzy rendszer komplex struktúrájában a használt tagfüggvények számán túl, azok felosztását is meg kell adni: a tagfüggvényeket a különböző bemeneti mezőkhöz rendeljük. A hozzárendelésben alapvetően célszerűbb több függvényt rendelni a szélesebb bemeneti értéktartománnyal rendelkező paraméterekhez. Jelen tanító halmazban a bemenet minden mezője diszkrét értékekből áll, melyek aránya 3:3:7:23, rendre a szemcseméret-átlag, szemcseméret-szórás, nyomtatási sebesség és AR bemeneti mezők sorrendjében. A kísérletek során így a várhatóan nagyobb futásidővel rendelkező tanításnál ezeket az arányokat közelítettük a tagsági függvények bemenetek közötti felosztásában. A tagsági függvények típusai közül a háromszög (Simpson) és a normális (Gauss) eloszlású függvényeket is vizsgáltuk – ezeket kizárólagosan alkalmaztuk egy-egy tanítás során.

A tagsági függvények számának, felosztásának és típusának megadása után a neuro-fuzzy rendszert a bemeneti adatok alapján tanítottuk a hálózat súlyainak megállapítására. A tanítás a súlyokat több iteráción keresztül hangolja. A több iteráció kisebb hibájú tanítást eredményezhet ám a tanítás futásideje lényegesen megnőhet, különösen nagyobb struktúrák esetén. Az iterációk számát ugyanolyan struktúrákra 5 és 25 között változtatva vizsgáltuk a tanítás hibáját, melyet az 5. ábra mutat. Az ábrán 4 különböző struktúra pontossági értékei szerepelnek a tagsági függvények felosztásának megjelölésével (pl. 2-2-4-6 a szemcseméret-átlag, szemcseméret-szórás, nyomtatási sebesség és AR bemeneti mezők sorrendjében).



5. ábra. MAPE értékek a *tanítási iteráció*k számának függvényében

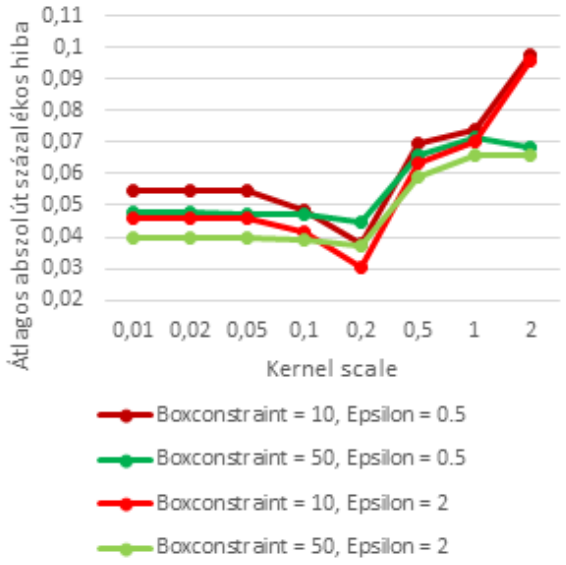
Megfigyeltük, hogy a hiba a komplexebb struktúrákban az iterációk növelésével csak kevéssé csökken, ugyanakkor a futásidő jelentősen nő. Mindezek mellett az átlagos MAPE értéke 19,9% lett optimális tanítással. Tehát a stencilnyomtatás folyamatára nem javasoljuk a neuro-fuzzy rendszerek használatát.

4.3. Szupport-vektor gépek

Ahogy a korábbi fejezetekben írtuk a szupport-vektor gépek többféle kernel funkciót is támogatnak. A számunkra ideális kernelfunkciót a tesztadatokra végzett néhány próba futtatással választottuk ki. A struktúra működési paramétereit a bemenet komplexitása alapján a szakirodalomban talált egyszerű heurisztikával határoztuk meg ehhez, így a *boxconstraint*, *epsilon* és *kernelscale* paraméterek értékeihez rendre a következőket definiáltuk: 23,9; 2,39 és 0,5. A linearis, polinomiális (négyzetes és köbös), valamint a Gauss-i kernelfunkciókkal a próbafuttatások során elért négyzetgyökös hibaátlaga (RMSE) rendre a következőképpen alakult: 16, 10, 8,8 és 8,3. Ez alapján a Gauss-i

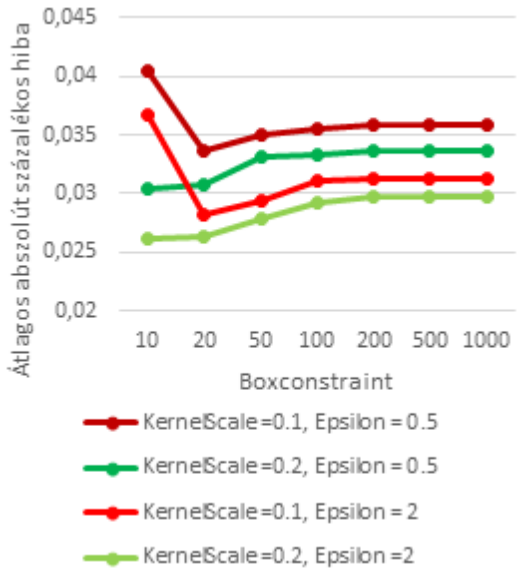
kernelfunkció további használata és a rá épülő modell részletes optimalizálása mellett döntöttünk.

A struktúra tanításának finomsága – és egyben gyorsaságának befolyásolása - a *boxconstraint*, *epsilon* és *kernel scale* paraméterek beállításával lehetséges. A *boxconstraint* határozza meg, hogy a kívülálló példányokra mekkora büntetést számolunk a tanítás során. Minél nagyobb ennek értéke annál finomabb modell képezhető, azaz a modell annál rugalmasabb, ugyanakkor könnyebben előfordulhat az úgynevezett túltanítás (overfitting). Az *epsilon* értéke megadja, hogy milyen értéknél kisebb hibát tekinthetünk nulla hibának a tanítás során. Minél kisebbre állítjuk, annál finomabb modell képezhető. Végezetül a *kernel scale* megadja, hogy az előrejelzés milyen finom skáláján változhat nagyobb mértékben a kernel funkció. Kisebb érték ismét finomabb modellt eredményezhet. A MAPE hibaértékek alakulását az egyes paraméterek függvényében a 6., 7. és 8. ábrákon mutatjuk.

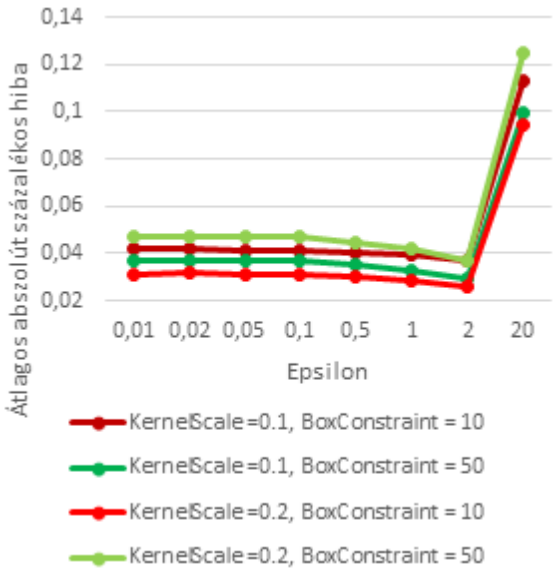


6. ábra. MAPE értékek a *kernel scale* paraméter függvényében

Megfigyelhető, hogy a tanítást befolyásoló paramétereknek van optimális értéke, igaz, hogy a pontosságban elérhető javulás néhol elég kicsi a paraméter változtatása mellett. A futásidőben jelentős különbségek adódhatnak, hiszen a finomabb struktúrák tanítási ideje lényegesen nagyobb lehet. Ugyanakkor a tapasztalt futásidők (5-10-15 másodperc) a gyakorlatban mind elfogadhatóak és könnyen kezelhetőek. Az egyes kimeneti értékek – terület, magasság és térfogat – előrejelzéséhez hasonló paraméterbeállítások adnak jó eredményeket, így található olyan beállítás, amelynél közel a teljes elméleti optimum is elérhető mindhárom kimenet esetére ugyanazokkal a tanítási beállításokkal.



7. ábra. MAPE értékek a *boxconstraint* paraméter függvényében

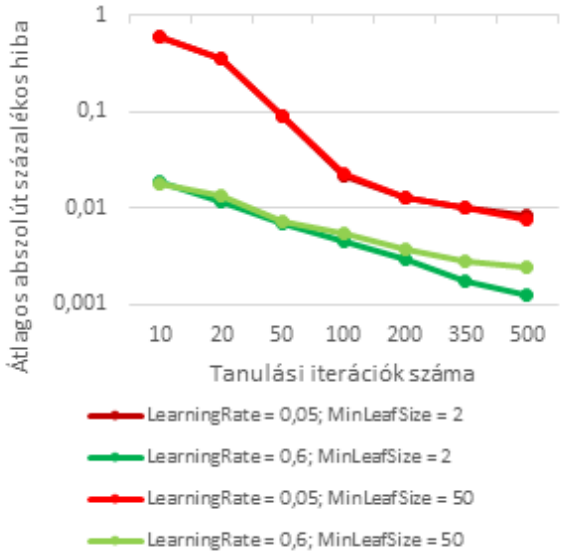


8. ábra. MAPE értékek az *epsilon* paraméter függvényében

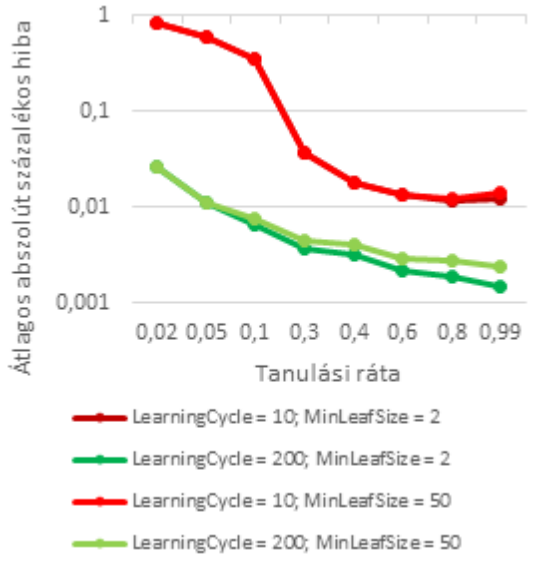
Az azonos paraméterbeállításokkal elérhető legjobb átlagos hiba 3,5%; az egyes kimenetekhez tartozó MAPE értékek 3%, 2,6% valamint 3,3% a terület, magasság és térfogat becslésére. A futásidő jelen beállítások mellett 5–6 másodperc körül mozgott a három kimenetre külön-külön. Összességében elmondhatjuk, hogy a szupport vektor gépek alacsony hibával képesek modellezni a stencilnyomtatás folyamatát.

4.4. Boost-olt döntési fák

A Boost-olt (vagy gyorsított) döntési fák egy hibrid modellt képviselnek a gépi tanulási módszerek között. A döntési fák egyszerűségét ötvözik robusztus tanulási módszerekkel a jobb eredmény elérése érdekében. A három paraméter, melynek függvényében a becslési hibát vizsgáltuk a *tanulási iteráció*k száma, a *tanulási ráta* és a *minimális levélméret*. Az egyes paraméterekre vonatkozó eredményeket a 9., 10. és 11. ábrák mutatják.

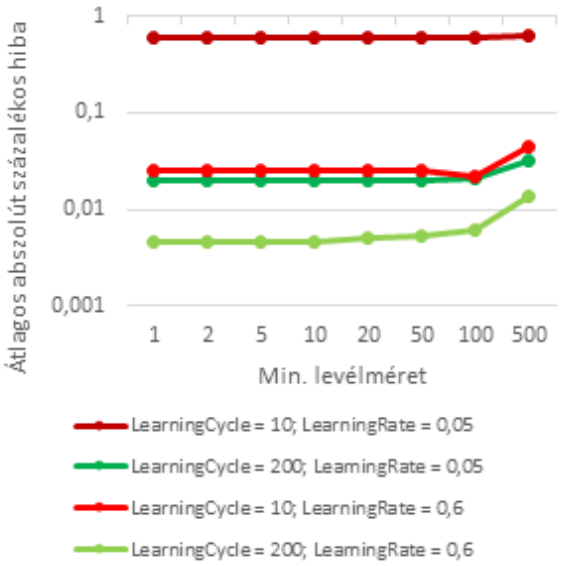


9. ábra. MAPE értékek a *tanulási iteráció*k számának függvényében



10. ábra. MAPE értékek a *tanulási ráta* függvényében

A teljes paramétertérben megfigyeltük, hogy az optimális beállítási paraméterek nagyon hasonlóan alakultak a három különböző kimenetre.



11. ábra. MAPE értékek a *minimális levélméret* függvényében

Megfigyeltük továbbá, hogy az optimális beállítások mellett a hibaértékek megközelíthetik az egy ezreléket is, nagyon kedvező kapcsolódó futásidők mellett (6–8 másodperc). Azt gyanítjuk, hogy paraméterek ilyen finom hangolásával a modellt túltanítottuk, ami a későbbi becslési képességét nagyban ronthatja hiányos adathalmazok esetére, amikor a tanításban részt nem vevő paraméterértékek becslését végezzük. Ezt kutatásunk következő fázisában tervezzük részletesebben vizsgálni. A túltanulást kiküszöbölve a boost-olt döntési fák előreláthatólag jól képesek modellezni a stencilnyomtatás folyamatát.

5. Összefoglalás

A vizsgált gépi tanulási módszerek teljesítményét nagyon jónak találtuk. Összevetve az előzetesen vizsgált lineáris regresszióval, aminek a becslési hibája a 100%-ot is elérte, a gépi tanulási módszerek akár hét nagyságrenddel is jobb megoldáshoz vezethetnek. Természetesen közöttük is vannak eltérések, így a neuro-fuzzy rendszerek nem bizonyultak megfelelőnek – legalábbis a többi megközelítéshez képest. A mesterséges neurális hálózatok, szupport-vektor gépek, valamint a döntési fák különböző változatai mind alkalmasak lehetnek a stencilnyomtatási folyamat kis hibájú modellezésére. Különbséget vélhetően a használhatóság során – annak bonyolultságában – tapasztalhatunk majd, melynek vizsgálata a kutatásunk következő szakaszára van tervezve.

Ennek megfelelően a kutatásunk következő szakaszában tervezzük a túltanulás jelenségének megfigyelését és kiküszöbölését az egyes módszerek tekintetében pl. hiányos tanítóhalmaz alapú tanítással és becsléssel. Tervezzük továbbá az eredmények kiértékelését a stencilnyomtatási folyamat technológiai mérőszámokkal való összevetésében, melyhez alkalmas – egyszerűen kezelhető – és többcélú (pl. milyen bemeneti értéktartományoknál lesz-lehet még éppen elfogadható a kimenet, vagy bizonyos rögzített működési paraméterek mellet a szabadon megválasztható egyéb paraméter-paraméterek értékét miképp állítsuk be, hogy a kimenet minél kedvezőbb legyen gyártási selejtszázalék minimalizálása céljából) megoldást kívánunk fejleszteni.

KÖSZÖNETNYILVÁNÍTÁS

A cikk a Bolyai János Kutatási Ösztöndíj támogatásával készült.

EMBERI ERŐFORRÁSOK MINISZTÉRIUMA

A cikk az Innovációs és Technológiai Minisztérium ÚNKP-20-5 kódszámú Új Nemzeti Kiválóság Programjának szakmai támogatásával készült.

IRODALOMJEGYZÉK

[1] Illés B., Géczy A., Skwarek A., Busek D., „Effects of substrate thermal properties on the heat transfer coefficient of vapour phase soldering”, *Int. J. Heat Mass Tran.*, 101. évfolyam, 69-75 o., 2016. *https://doi.org/10.1016/j.ijheatmasstransfer.2016.04.116*

[2] Illés B., Géczy A., „Investigating the heat transfer on the top side of inclined printed circuit boards during vapour phase soldering”, *Appl. Therm. Eng.*, 103. évfolyam, 1398-1407 o., 2016. *https://doi.org/10.1016/j.applthermaleng.2016.04.153*

[3] Lau C.S., Khor C.Y., Soares D., Teixeira J.C., Abdullah M.Z., „Thermo–mechanical challenges of reflowed lead–free solder joints in surface mount components: a review”, *Solder. Surf. Mount Technol.*, 28. évfolyam, 2. szám, 41-62 o., 2016. *https://doi.org/10.1108/SSMT-10-2015-0032*

[4] Skwarek A., Synkiewicz B., Kulawik J., Guzdek P., Witek K., Tarasiuk J., „High temperature thermogenerators made on DBC substrate using vapour phase soldering”, *Solder. Surf. Mount Technol.*, 27. évfolyam, 3. szám, 125-128 o., 2015. *http://dx.doi.org/10.1108/SSMT-04-2015-0017*

[5] Tsai T.N., „Modeling and optimization of stencil printing operations: A comparison study”, *Comput. Ind. Eng.*, 54. évfolyam, 3. szám, 374-389 o., 2008. *https://doi.org/10.1016/j.cie.2007.08.001*

[6] Pan J., Tonkay G.L., Storer R.H., Sallade R.M., Leandri D.J., „Critical Variables of Solder Paste Stencil Printing for Micro–BGA and Fine–Pitch QFP”, *IEEE T. Electron. Pa. M.*, 27. évfolyam, 2. szám, 125-132 o., 2004. *https://doi.org/10.1109/TEPM.2004.837965*

[7] Hastie T., Tibshirani R., Friedman J.H., „The Elements of Statistical Learning”, *Springer-Verlag*. 2009. ISBN: 978-0-387-84858-7

[8] MacKay D.J.C., „Bayesian interpolation”, *Neural Comput.*, 4. évfolyam, 415-447 o., 1992. *https://doi.org/10.1162/neco.1992.4.3.415*

[9] Foresee F.D., Hagan M.T., „Gauss-Newton approximation to Bayesian regularization”, *Proceedings of the 1998 International Joint Conference on Neural Networks*, 1930-1935 o., 1997. *https://doi.org/10.1109/ICNN.1997.614194*

[10] Hagan M.T., Menhaj M.B., „Training feedforward networks with the Marquardt algorithm”, *IEEE Trans. Neural Netw.*, 5. évfolyam, 989-993 o., 1994. *https://doi.org/10.1109/72.329697*

[11] Møller M.F., „A scaled conjugate gradient algorithm for fast supervised learning”, *Neural Netw.*, 6. évfolyam, 525-533 o.,1993. *https://doi.org/10.1016/S0893-6080(05)80056-5*

[12] Breiman L., Friedman J.H., Olshen R.A., Stone C.J., „Classification and regression trees”, *Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software*. 1994. ISBN 978-0-412-04841-8.

[13] Friedman J.H., „Stochastic Gradient Boosting” (March 1999), elérve: *https://statweb.stanford.edu/~jhf/ftp/stobst.pdf 2021*. március 9.

[14] Jang J.-S.R. „ANFIS: Adaptive-Network-Based Fuzzy Inference System”, *IEEE T. Syst. Man Cyb.*, 23. évfolyam, 3. szám, 665-685 o., 1993. *https://doi.org/10.1109/21.256541*

[15] Krammer O., Al-Ma’aiteh T., Martinek P., Anda K., Balogh N., „Predicting the Transfer Efficiency of Stencil Printing by Machine Learning Technique”, *2020 43rd International Spring Seminar on Electronics Technology (ISSE)*, 1-6. o., 2020. *https://doi.org/10.1109/ISSE49702.2020.9121032*