Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Bemenet optimalizáció neurális generatív modellek értelmezéséhez

Freund László
szerző
Budapesti Műszaki és
Gazdaságtudományi Egyetem

Hadházi Dániel
konzulens
Budapesti Műszaki és
Gazdaságtudományi Egyetem

Dr. Orbán Gergő
konzulens
Wigner Fizikai
Kutatóközpont

2023, Tudományos Diákköri Konferencia

Kivonat

A neurális modellek egyre fontosabb szerepet játszanak nem csak a gépi tanulás, de az idegtudomány területén is, ahol többek között a látókéreg működésének vizsgálatára, illetve a modellezésére használják őket. Probléma azonban, hogy ezen modellek mélyebb réteginek tanult reprezentációi még mindig rejtélyesek és nehezen vizsgálhatóak. Ezeknek az értelmezhetősége nagy jelentőséggel bír, mivel így átláthatóbb és megbízhatóbb rendszereket tudunk tervezni, vagy akár az agyi folyamatok megértését is elősegíthetjük.

Egy lehetséges módszer a moelljeink viselkedésének értelmezésére az Activation Maximization (AM), amely olyan bemeneteket generál, amelyek maximalizálják egy adott egység vagy egységcsoport aktivitását. Ezzel feltárhatjuk, hogy milyen tulajdonságokra érzékenyek a modelljeink belső egységei. Az AM módszernek (és általánosságban véve a bemenet optimalizációs algoritmusoknak) azonban vannak korlátai, ugyanis regularizáció nélkül gyakran olyan mintákat hoznak létre, amelyek nem tartoznak a neurális eszköz által modellezett és tanult háttéreloszlásba, így nem reprezentálják jól a mintákat jellemző és megtanult tulajdonságokat. Ebben a munkában arra keresek váltaszt, hogy miként lehet a problémát a gyakorlatban kezelni.

Megvizsgálom a szakirodalomban leírt fontosabb módszereket, majd javaslatot teszek ezek adaptációjára, és továbbfejlesztésére. Kutatásom során sikerült olyan kombinált megközelítést is javasolnom, mely alkalmazása esetén jó minőségű, hiperparaméterrel hangolható tulajdonságokkal rendelkező, a vizsgált háló aktivációinak reprezentációit szemléletesen leíró, magyarázó bemeneteket tudok generálni. Javaslatot teszek továbbá a generatív modellek jellemzőihez jól illeszkedő megközelítésekre is. Munkám során kidolgozott módszereket kvalitatívan és kvantitatívan is minősítem, melyhez szintén lehetséges megközelítést javaslok. Továbbá ezen minősítések során értelmezem is a tapasztaltakat, hippotéziseket fogalmazok meg és tesztelek vissza.

A vizsgálathoz egy hierarchikus Variational Autoencoder (VAE) modellt használok, amely természetes textúrák rekonstruálására tanítottak, és alkalmas a korai látókéreg működésének modellezésére [4]. Ez alapján a modell látens paraméterei érzékenyek a bemeneti képben bizonyos textúrák mintázatainak jelenlétére. A dolgozatban bemutatott módszereket ezen paraméterek által tanult reprezentációk elemzésére használom.

Abstract

Neural models play an increasingly important role not only in machine learning but also in neuroscience, where these models have been instrumental in the exploration and modeling of various aspects, including the functioning of the visual cortex. However, the learned representations of the deeper layers of these models are still mysterious and difficult to examine. The interpretability of these representations is of great significance, as it allows us to design more transparent and reliable systems or even facilitate the understanding of neural processes.

A possible method for interpreting the behavior of our models is Activation Maximization (AM), which generates inputs that maximize the activity of a given unit or group of units. This way, we can reveal what properties our models' internal units are sensitive to. However, the AM method (and generally speaking, all similar input optimization algorithms) have limitations. Without regularization these techniques tend to create samples that are inconsistent with the underlying distribution learned by the neural model, thus they do not represent well the characteristics and learned properties of the latent features. In this work, I investigate how to handle this problem in practice.

In my research, I examine the key methodologies described in the existing literature and subsequently propose recommendations for their adaptation and enhancement. I have also managed to suggest a novel approach, which allows for the generation of high-quality visualizations which can vividly describe the representations of learned by neural networks. This methodology offers the unique capability of controlled manipulation over the salient attributes of the generated samples.

Moreover, I put forth suggestions for approaches that are based on the principles of generative models. I evaluate the methodologies I have developed both qualitatively and quantitatively. During these evaluations, I interpret the findings, formulate hypotheses, and conduct tests to validate them.

We use a hierarchical Variational Autoencoder (VAE) model that we trained to reconstruct natural texture images and is suitable for modeling the functioning of the early visual cortex [4]. Accordingly, the model's latent parameters exhibit sensitivity to the presence of specific texture patterns in the input image. The methods presented in this paper are utilized for the analysis of representations learned by these parameters.

1. Bevezetés

Az emberi agy működésének a megértése az emberiség történelmének hosszú ideje tartó törekvése. Az agy megértésének egyik útja az, hogy tanulmányozzuk mit kódol egy-egy neuron vagy neuron-csoport [11], vagyis milyen információt reprezentál az aktiválódásuk. A klasszikus 1950-es évekbeli kísérletben Hubel és Wiesel egy macska agyát tanulmányozták úgy, hogy a vizsgálati alany számára különböző képeket mutattak egy képernyőn, miközben rögzítették a macska elsődleges látókéregében lévő neuronok ingerlését. A sokféle tesztkép közül a kutatók azt találták, hogy az orientált élek erős választ váltanak ki adott sejtekből [10]. Ezeket a sejteket él-detektoroknak nevezik, és az ilyen képeket preferált ingereknek hívják.

Ugyanez a technika később lehetővé tette, hogy alapvető eredményekre jussunk arról, hogy a látópálya mentén lévő neuronok miként érzékelnek egyre bonyolultabb mintázatokat: köröktől, élektől az arcokig és magas szintű fogalmakig, mint bizonyos családtagok vagy hírességek [35].

Hasonlóképpen, a gépi tanulásban a neurális egységek preferált ingereinek vizuális vizsgálata világosabb magyarázatot adhat az adott neuron viselkedésére [42] [43] . Egy intuitív megközelítés, hogy megtaláljuk az ilyen preferált bemeneteket egy létező, nagy képgyűjteményből pl. a tanító vagy teszt halmazból [43]. Ez a módszer azonban nem kívánt tulajdonságokkal rendelkezhet: (1) megköveteli, hogy minden neuront teszteljünk egy nagy képkészleten; (2) egy ilyen adathalmazban sok olyan informatív kép lehet, amely aktiválná az egységet, de nem létezik, mert a képtér óriási, és a neurális viselkedések bonyolultak lehetnek [23]; (3) gyakran nem egyértelmű, hogy melyik vizuális jellemző váltja ki a neuron tüzelését pl. ha egy egység aktiválódik egy madár képétől egy faágon, nem világos, hogy az egység a madár vagy az ág iránt „érdeklődik-e”; (4) nem triviális, hogyan vonjunk le egy holisztikus következtetést arról, hogy mit tanult meg egy neuron a tipikusan nagy méretű ingerek halmazából, amelyeket a neuron preferál. Egy gyakori megközelítés az, hogy tanulmányozzuk a legmagasabb aktivitást mutató kilenc képet egy egységhez [43]. A top 9 halmaz azonban nem feltétlenül tükrözi egyértelműen az egység által preferát jellemzőt [24].

Egy létező adathalmazból való képek megtalálása helyett a semmiből is lehet szintetizálni a vizuális ingereket [6]. A szintézis alapú megközelítés több előnnyel is jár: (1) erős kép prior ismeretével a vizsgált modell gyakorlatban nem feltétlen elérhető tanítóhalmazához való hozzáférés nélkül lehet mintákat szintetizálni; (2) több irányításunk van a szintetizálni kívánt képek típusai és tartalmai felett, ami kontrolláltabb kutatási kísérleteket tesz lehetővé, pl. annak vizsgálatát, hogy több neuron hogyan reprezentál közösen információt; (3) elválasztja azokat a tulajdonságokat, amelyek a viselkedést okozzák azoktól, amelyek csak korrelálnak az okokkal.

A neurális hálózatok kézenfekvő tulajdonsága, hogy differenciálhatóak a bemenet szerint. Ha egy olyan bemenetet szeretnénk találni, ami egy bizonyos viselkedést vált ki a modellből, lehet ez a modell kimenete, de lehet a rejtett rétegek neuronja is, akkor ezeket a deriváltakat használhatjuk annak érdekében, hogy iteratívan a cél felé mozgassuk a bemenetet. A backpropogation algoritmus egy egyszerű bővítéssel, a visszaterjesztés egészen a bemenetig való folytatásával alkalmazható erre a feladatra. Ezt a módszert a szakirodalom aktiváció maximalizálásnak (Activation Maximization, AM) nevezi [6].

1. ábra. A GoogLeNet [38] négy egységét legjobban aktiváló 9 kép az ImageNet adathalmazból, illetve az egységek szintetizált preferált bemenetei. Forrás: [27]

A koncepció egyszerűsége ellenére a gyakorlatban ez egy összetettebb probléma, ugyanis ez a módszer ”optikai illúziókat” hoz létre, amelyek tele vannak zajjal és értelmetlen magas frekvenciás elemekkel, de magas aktivációt váltanak ki a modellből. Ezek gyakran nem tartoznak a neurális eszköz által modellezni kívánt adatok háttéreloszlásába, így nem reprezentálják jól a mintákat jellemző és megtanult tulajdonságokat. Ezt az optimalizációs folyamatot egyféle ”csalásként” is felfoghatjuk, így közeli összefüggésbe hozható az ellenséges (adversarial) mintákkal [39] [23]. Ezek a ”csaló” képek magukban is érdekesek, mivel megértésükkel modelljeink robosztusabbá tehetőek az ilyen minták ellen. Ha azonban arról szeretnénk többet megtudni, hogy miként viselkedik a modellünk a tanult eloszlásán belül, akkor megkötéseket kell tegyünk az optimalizációba. A területen a kutatás jelentős része egy olyan megkötés keresése, amivel minél közelebb terelhetjük az optimalizált bemenetet a természetes képek eloszlásához. Természetes képek alatt a dolgozat további részeiben olyan eloszlás mintavételeit értem, melyek fizikai valóságban létező objektumok látható fotonokon alapuló képrögzítésével keletkeznek. A természetes képeken belül emberileg értelmezhetőnek hívom azokat a képeket, melyek koherens, nem csak zaj-szerű (alacsonyabb entrópiájú) struktúrákat tartalmaznak.

1.1. Regularizáció

A szakirodalmat az optimalizációs kényszer választása alapján három kategóriába tudjuk sorolni:

- a **magas frekvenciájú komponens alapú** módszerek egyenesen a magas frekvenciás komponensek eltüntetését célozzák meg. Ez a gyakorlatban:
  - a képen valamilyen zajtalanító folyamat futtatása minden iterációban [23];
  - zaj büntetése a bemenet optimalizációjának költségfüggvényében (pl. a bemenet totalális varianciájának redukálásával) [18];
  - zajtalanítás végzése az optimalizáció során kiszámított gradiensekben (pl. aluláteresztő szűrő alkalmazásával) [44].

  A hátránya ennek a megközelítésnek, hogy azokat a magas-frekvenciás komponenseket is eltüntetik, amelyek lényeges részei lehetnek a tanult reprezentációnak.

– a **transzformációs robosztusság alapú** módszerek olyan mintákat keresnek, amelyek akkor is aktiválják a választott egységet, ha egy enyhe transzformációt végzünk rajtuk. A gyakorlatban ez a bemeneti képre alkalmazott iterációnkénti sztochasztikus jitter-elést, nagyítást, forgatást vagy egyéb stabilitást kikényszerítő transzformációt jelent [20] [44].

– a **tanult prior eloszláson alapuló** módszerek az előzőekkel ellentétben nem egyszerű heurisztikákat használnak annak érdekében, hogy ésszerű mintákat generáljanak. Egy kézenfekvő következő lépcsőként, valódi adatokra megtanított modellt alkalmaznak, és az ehhez való igazodást próbálják kikényszeríteni a generált mintákból. Ezen módszerek eredményezik a leginkább foto-realisztikus mintákat, azonban hátrányuk, hogy nem lehet megmondani melyik információ származik ténylegesen a modellből és melyik a külön megtanult priorból. A gyakorlatban ezt egy olyan generatív modell építésével érik el, amely egy látens térben lévő pontokhoz tanul meg valódi adatpont párokat, és ebben a látens térben végzi az optimalizációt [22]. Egy másik megközelítésben egy olyan priort tanítanak meg, ami eléri a valószínűség gradiensét, így egyszerre lehet optimalizálni a priorra és a cél aktivációra [21].

1.2. Változatosság

Az optimalizáció alapú minta-generálásnak egy másik buktatója, hogy az adott neuron kimenetét egy eloszlás mintái aktiválják erősen, és ebből az eloszlásból csak egy mintavétel alapján nem tudunk az eloszlás általános jellemzőire következtetni. Tehát nem tudjuk, hogy amit eredményül kaptunk lefedi-e a teljes képet, vagy csak egy aspektusból mutatja meg, hogy mit reprezentálnak az adott tulajdonságok. Jóllehet az optimalizáció eredménye általánosságban egy erős aktivációt kiváltó minta, de nem tudjuk megmondani, hogy ennek melyik része az, ami konkrétan kiváltja ezt a viselkedést. Több egymástól különböző minta generálásával viszont megfigyelhetőek az ezek közti különbségek és hasonlóságok, ezzel szétválasztva, hogy melyek azok a komponensek, amelyek csak egy adott optimalizációra igazak, és melyek azok, amelyek ténylegesen kiváltják a keresett viselkedést. Igény van tehát egy feladaton belül egymástól eltérő, de egyenként is értelmezhető minták előállítására. Erre több megközelítést is vizsgál a szakirodalom:

– Az optimalizáció kiindulási pontjának a változtatása. Ezek a kezdeti minták lehetnek az adathalmaz szélsőséges pontjai [24], vagy akár osztályozási feladat esetén egy osztály adatpontjainak a közepe.

– Egy változatossági tényező bevezetése az optimalizációs feladatba, amely több formában is megjelenhet. Ezeknek a hátrányait és előnyeit még nem vizsgálták átfogóan a szakirodalomban. Egy megközelítés lehet például a minták közötti koszinusz hasonlóság büntetése [27], de olyan próbálkozások is voltak, amelyek a stílus transzfer területéről inspirálódva különböző stílus priorokba akarják kényszeríteni a mintákat [8].

– Az adathalmazon megtanított priorral rendelkező generatív modellekkel is elérhetőek változatos eredmények mindössze a modellből való mintavételezéssel [22].

Változatos minták generálásával még pontosabban meg tudjuk határozni, hogy melyek azok a jellemzők, amelyek adott aktivációkat kiváltanak, akár addig a szintig, hogy az adathalmazunkból példákat nézve meg tudjuk jósolni ezen aktivációkat [27]. A generált

minták változatos irányokba kényszerítésének hátránya, hogy megjelenhetnek nem oda tartozó komponensek is, tehát a generált minták magukban kevesebb relevanciával bírnak az adott feladatra. Itt megjelenik egy még fundamentálisabb probléma. Jóllehet könnyű lenne feltételezni, hogy adott neuronok összefüggő fogalmakat reprezentálnak, azonban a gyakorlatban megjelennek olyan egységek is, amelyek különböző dolgok egy furcsa elegyének a felismerését tanulták meg. Az ilyen minták felvetik a kérdést, hogy vajon az egyedülálló neuronok a jó szemantikus egységek-e a neurális hálózatok megértéséhez [27].

1.3. Megközelítés

A generatív modellek definíciójuk szerinti feladata egy adathalmaz mögötti háttéreloszlás megtanulása. Ennek gyakorlati következménye, hogy ezen modellek lehetővé teszik egy bemenet valószínűségének, vagy ezen valószínűség egy közelítésének a kiszámítását. Célunk olyan minták létrehozása, amelyek beletartoznak a modellünk által megtanult eloszlásba, így a regularizáció az ezen valószínűség magasan tartásával természetes megoldást nyújt a generált mintáknak a modell priorjába kényszerítésére. Ezentúl a generatív modellek képesek az általuk megtanult eloszlásból mintavételezve az ebbe az eloszlásba tartozó új képek szintetizálására. Munkámban ezeket a tulajdonságokat használom ki annak érdekében, hogy mélyebb rálátást nyerjek a modell egyes egységeinek a működésébe. Ennek a megközelítésnek az előnye, hogy a generálás folyamatához nincs szükségünk egy újabb modell betanítására az adathalmazon, tehát biztosak lehetünk abban, hogy a generált minta minden információt a vizsgált modellből inferált. Továbbá új módszereket javaslok a szintetizált minták minőségének és értelmezhetőségének a javítására: a pusztán egy kép pixel szintű optimalizálásán túl bevezetek két új módszert. (1) Egy eloszlás paraméterein való optimalizációt, amely lényegesen gyorsabb konvergenciát és egyes esetekben letisztultabb mintákat eredményez; valamint (2) a paraméterek egy tanulható transzformációval (pl. konvolúció, neurális hálózat) történő bővítését, amely a korábbi módszereknél jobb minőségű, magasabb aktivációt implikáló eredményekre jut, valamint lehetővé teszi a szintézis eredményének intuitív irányítását. Ezen módszerek vizsgálatához egy hierarchikus Variational Autoencoder (VAE) modellt használok, amely természetes textúrák rekonstruálására alkalmas, és amely működésében hasonlóságokat mutat a korai látókéreg által elvégzett funkciókkal [4]. A munkámmal segíteni szeretném a látókéreg működésének mélyebb absztrakciókon történő értelmezését, azonban a dolgozatban nem célom a preferált bemeneteken vizualizált reprezentációk részletes értelmezése, továbbá a neurális modell és az általa modellezett rendszer közötti esetleges eltérések feltérképezése sem.

1.4. Felépítés

A dolgozatban formálisan is definiálom a problémát általánosítva a létező regularizációs technikákra (2. fejezet). Betekintést adok a generatív modellek, azon belül is a Variational Autoencoder modell, majd részletesebben a konkrét vizsgált modell működésébe (2.5. fejezet). Bemutatom az elvégzett kísérleteket (3. fejezet), majd az eredmények részletezése és megvitatása (4. fejezet) után összegzem a leírtakat (5. fejezet).

2. Háttér

2.1. Aktiváció maximalizálás (AM)

Jelölje θ egy neurális hálózat paramétereit, amely egy $x \in \mathbb{R}^{H \times W \times C}$ képet (amelynek C szín csatornája van, mindegyik W pixel széles és H pixel magas) kap bemenetként. Egy olyan x kép megtalálása, amely maximalizálja az $a(l,i)(\theta\, x)$ aktivációt egy adott l rétegben indexelt i neuronban, megfogalmazható optimalizálási problémaként:

$$x^* = \arg\max_x \left[a(l,i)(\theta, x)\right]$$

Ezt a problémát Erhan, Bengio és kutatótársaik aktivációs maximalizálásnak (AM) nevezték el [6]. Itt az $a(l,i)$ az egyes egységek aktivációs értékét adja vissza, azonban $a(.)$ bármilyen, tanulmányozni kívánt neurális válaszra kiterjeszthető, amelyet tanulmányozni szeretnénk (pl. egy csoport neuron aktiválása) [20] [28]. A figyelemre méltó DeepDream vizualizációkat [20] úgy hozták létre, hogy AM-et futtattak az adott réteg összes egységének egyidejű aktiválására. (Ettől a ponttól kezdve a-t fogok írni $a(l,i)$ helyett, amikor a pontos indexeket l, i elhagyhatjuk az általánosság kedvéért.) Az AM egy nem-konvex optimalizálási probléma, amelyre kereshetünk egy lokális maximumot gradiens-alapú [39], vagy nem gradiens alapú módszerekkel [25]. A hasonló feladatoknál gyakran feltételezzük a tanulmányozott hálózat paramétereinek és architektúrájának elérhetőségét. Ebben az esetben egy egyszerű megközelítés a gradiens emelkedés [6] [42] végrehajtása egy olyan frissítési szabállyal, mint [26]:

$$x(t+1) = x(t) + \epsilon_1 \cdot \frac{\partial a(x(t))}{\partial x(t)}$$

Azaz kiindulva egy véletlenszerű $x(0)$ inicializálásból (itt egy véletlenszerű kép), iteratívan lépéseket teszünk a bemeneti térben követve az $a(x)$ gradiensét annak érdekében, hogy megtaláljunk egy x bemenetet, amely nagymértékben aktiválja az adott egységet. ϵ_1 a lépésméret, melyet empirikusan választunk meg. Vegyük észre, hogy ez a gradiens emelkedési folyamat hasonló a neurális hálózatok tanításához használt gradiens csökkenési folyamathoz, a backpropagation [36] algoritmushoz, kivéve, hogy itt mi a hálózat bemenetét optimalizáljuk a hálózat paraméterei helyett, amelyek rögzítettek. Az optimalizálást leállíthatjuk, amikor a neurális aktiváció elérte a kívánt küszöböt, vagy meghatározott számú lépés telt el. A gyakorlatban egy kép szintetizálása csak az aktiváció maximalizálásával (egy korlát nélküli optimalizálási probléma) gyakran értelmezhetetlen mintákat eredményez [23]. Egy nagydimenziós képtérben gyakran találunk megtévesztő/"szemét" példákat pl. magas frekvenciájú zaj mintákat, amelyek semmire sem hasonlítanak, de nagymértékben aktiválják az adott egységet. Egy valós képről kiindulva (egy véletlenszerű helyett) azonban könnyen találkozhatunk ellenséges (adversarial) példákkal [39] (pl. egy olyan kép, amely kicsit eltér a kiinduló képtől, viszont a hálózat teljesen más címkét rendel hozzá). Ezek a korai AM vizualizációk óriási biztonsági és megbízhatósági problémákat tártak fel a gépi tanulás alkalmazásokkal kapcsolatban és számos követő ellenséges támadási és védekezési kutatást indítottak el.

2.2. Kézzel tervezett prior alapú AM regularizáció

A tény, hogy a hálózat erősen reagál "szemét" példákra, érdekes és erős következményekkel jár a modell biztonsága és robosztussága szempontjából. Azonban, ha nem tudjuk értelmezni a szintetizált képeket, az korlátozza a képességünket arra is, hogy megértsük, mi az adott egység célja. Ezért azt akarjuk, hogy a keresést egy olyan képeloszláson belülre korlátozzuk, amelyet értelmezni tudunk (pl. fotórealisztikus képek, vagy olyan képek, amelyek hasonlítanak a tanítókészletben lévőkre). Ezt úgy lehet elérni, hogy természetes kép regularizációt építünk be a célfüggvénybe, amely jelentősen javítja az AM képek felismerhetőségét [18][27][42]. Ehhez MAP (Maximum a posteriori estimation) becslést alkalmazunk a következőképpen:

$$x^* = \arg\max_x \left(p(x|a)\right) = \arg\max_x \left(p(a|x) \cdot p(x)\right)$$

Például egy kép prior simaságot ösztönözhet [18], vagy büntetheti a rendkívüli intenzitású pixeleket [27]. Az ilyen korlátokat gyakran beépítik az AM formulába egy regularizációs tagként, az előző képlet logaritmusát véve [26]:

$$R(x) : x^* = \arg\max_x \left[a(x) - R(x)\right]$$

Annak érdekében, hogy ösztönözzük a simaságot az AM képeken, $R_{\text{obj}} : \mathbb{R}^{H\times W\times C} \to \mathbb{R}$ kiszámíthatja a totális varianciát (Total Variation, TV) egy képen [18]. De más regularizációs költséget is bevezethetünk, például a túlzott intenzitású pixelek büntetését α-normával [37]. Azaz minden frissítésben úgy módosítjuk a bemenetet, hogy (1) maximalizljuk a neurális aktivációt és (2) minimalizáljuk a regularizációs büntetést:

$$x(t+1) = x(t) + \epsilon_1 \frac{\partial a(x(t))}{\partial x(t)} - \epsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)}$$

A gyakorlatban azonban nem mindig számoljuk ki a $\partial R_{\text{obj}}(x(t))/\sigma x(t)$ analitikus gradienst. Ehelyett definiálhatunk egy $R_{\text{post}} : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times C}$ regularizációs (proximal) operátort [31], és x-et minden optimalizációs lépés után egy jobban regularizált (pl. kissé homályosabb [42]) változatára térképezzük. Ezzel is a magas frekvenciájú zajt büntetjük valamilyen zajtalanítási módszerrel, például teljes variáció zajtalanítással [18], Gauss-elmosással [42][44] vagy bilaterális szűrővel [40]. Ebben az esetben a frissítési lépések azonos értékekhez konvergálnak, mint gradiens emelkedésnél, és a következőképpen alakulnak:

$$x(t+1) = R_{\text{post}}\left(x(t) + \varepsilon_1 \frac{\partial a(x(t))}{\partial x(t)} - \varepsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)}\right)$$

Az optimalizált minta zajérzékenysége redukálható az optimalizációs lépés előtt egy $R_{\text{pre}} : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times C}$ regularizációs elem bevezetésével. Ez olyan transzformáció lehet, amely nem szakítja meg a backpropogation algoritmus gradiens-áramlatát. Véletlenszerűen rázzuk meg, forgassuk el, vagy méretezzük át a képet minden frissítési lépés előtt, hogy

olyan bemeneteket szintetizáljunk, amelyek által kiváltott ingerek robusztusan tűrik ezen transzformációkat. Ennek következtében az eredmények tisztábbak és értelmezhetőbbek [20] [27] lesznek. Kiegészítve így alakul a frissítési szabály:

$$x(t+1) = R_{\text{post}}\left(x(t) + \varepsilon_1 \frac{\partial a\left(R_{\text{pre}}(x(t))\right)}{\partial x(t)} - \varepsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)}\right)$$

Egy másik jól bevált regularizációs technika a gradienskép $\partial\, a(x(t))/\partial\, x(t)$ magas frekvenciáinak büntetése (a vizualizáció $x(t)$ helyett) szűréssel. Ehhez egy $R_{\text{grad}} : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times C}$ regularizációs operátort vezetünk be. Ez gyakorlatban egy aluláteresztő szűrő (Gauss-elmosás) [27][44] alkalmazását jelenti a gradiensen. Általánosítva tehát így formalizálhatóak a taglalt regularizációs eljárások:

$$x(t+1) = R_{\text{post}}\left(x(t) + R_{\text{grad}}\left(\varepsilon_1 \frac{\partial a\left(R_{\text{pre}}(x(t))\right)}{\partial x(t)}\right) - \varepsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)}\right)$$

Jóllehet jelentősen javítják a képek értelmezhetőségét (a magas frekvenciás zaj által dominált példákhoz képest), ezek az eljárások csak hatékonyan próbálnak megfelelni a természetes képek átlalános lokális statisztikáinak. Ez azonban nem mindig elegendő.

2.2.1. Globális struktúrák

Sok AM képből még mindig hiányzik a globális koherencia [26]; például egy olyan kép, amelyet úgy szintetizáltak, hogy erősen aktiválja a „paprika" kimeneti neuronját, több paprika-szeletet is mutathat ugyanazon a képen egy paprika helyett. Ilyen ingerek arra utalnak, hogy a hálózat megtanult néhány helyi diszkriminatív jellemzőt, pl. a paprikák fényes, zöld felületét, amely hasznos a besorolási feladathoz. Ez felvet egy érdekes kérdést: vajon a hálózat valaha megtanulta a globális struktúrákat (pl. az egész paprikát), vagy csak a helyi diszkriminatív részeket? Amikor pixelenként változtatjuk meg a képet, nem triviális biztosítani a globális koherenciát annak egészén. Ehelyett könnyű növelni a neurális aktivációkat egyszerűen több helyi diszkriminatív jellemző létrehozásával az ingerben. A globális koherencia javításának korábbi kísérletei közé tartoznak az alábbi megközelítések:

- Szabályozzuk a képváltozásokat a generált kép közepéhez közel [24].
- Inicializáljuk az optimalizációt egy átlagos képről (a valódi tanítóhalmaz képeiből számítva) egy véletlen helyett [24].

Bár ezek az eljárások némileg javították a képek globális koherenciáját, ezek heurisztikákon alapulnak és extra, nehezen hangolható hiperparaméterek bevezetését követelik [44][24]. Ráadásul még mindig nagy az eltérés a valódi képek és ezen vizualizációk között.

2.2.2. Változatosság

Egy neuron többoldalú lehet abban az értelemben, hogy erősen reagál több különböző típusú ingerre, azaz aspektusokra [24]. A magasabb szintű jellemzők invariánsabbak a

bemenet változásaihoz képest [43]. Például a híres CaffeNet [15] egy-egy arc-érzékeny egységében azt találták, hogy mind az emberi, mind az oroszlán arcokra reagál [42]. Sokszor különböző aspektusokat érdemes ezért felfedni AM segítségével annak érdekében, hogy jobban megértsük egy-egy vizsgált egység viselkedését.

Az AM optimalizáció azonban különböző véletlen képekből indulva gyakran hasonló eredményekre konvergál [6][24] - egy jelenség, amelyet akkor is megfigyeltek, amikor neurális hálózatokat különböző inicializációkkal tanítanak [17]. A kutatók különböző technikákat javasoltak a képdiverzitás javítására, pl.:

- Rendezzük a tanítóhalmaz képeit csoportokba, és inicializáljunk egy, a különböző csoportokból kiszámított átlagos képből [25].
- Maximalizáljuk a távolságot (pl. koszinusz hasonlóság a pixel-térben) egy referenciakép és a szintetizált kép között [27].
- Aktiváljunk több neuront egyszerre. Pl. aktiválva a (madár + kutya) és a (madár + macska) felismerését végző egységeket, így két különböző madárképet állítunk elő, amelyek aktiválják ugyanazt a madár egységet [22].
- Adjunk zajt a képhez minden frissítésnél, hogy növeljük a képdiverzitást [21].

Ezek az eljárások is korlátozott sikerrel jártak, mégis extra hiperparamétereket vezetnek be, amelyek további vizsgálatot igényelnek. Például, ha azt szeretnénk, hogy két inger különbözzön, pontosan milyen messze kell lenniük, és milyen hasonlósági mértékben kell mérni a különbséget?

2.3. Tanult prior eloszláson alapuló AM regularizáció

A kézzel tervezett prior alapú optimalizációnál megjelenő R_{obj} célfüggvény regularizációs tag célja, hogy a mintát $p_{\text{gt(x)}}$ közelébe terelje. Ezt ott $p_{\text{gt(x)}}$ egy általános természetes priorral való közelítésével próbálja megvalósítani (pl. Total Variation esetén a gradiens mező ritkaságát feltételező - Compressed Sensing megközelítés által indukált priort approximáljuk). Az adathalmazunk mögötti háttéreloszlásnak azonban ez gyakran nem elég jó becslése jó minőségű minták generálásához. Erősebb prior definiálásával azonban egy specifikusabb kényszert tudnánk adni az optimalizációnak. Ennek a priornak a megtalálására egy lehetséges út egy generatív modell betanítása [21]:

$$p_{\text{gen}}(x) \approx p_{\text{gt}}(x)$$
$$R_{\text{obj}}(x) = \varepsilon_1 \times p_{\text{gen}}(x)$$

A generatív neurális hálózatokban a bemenetre visszaterjeszthető a vizsgált aktiváció gradiense, így lehetőséget nyújtanak arra, hogy egyszerre optimalizáljunk az általuk definiált priorra és a vizsgált egység aktivációjára [21]. Ez a regularizációs megközelítés nagyot javít a képminőségen, azonban igényli egy másodlagos neurális architektúra definiálását és betanítását az adathalmazon. Mivel ezen másodlagos architektúra értelmezése ugyanolyan nehézségeket vet fel, mint a vizsgált modellünké, így nem tudjuk megmondani, hogy a szintetizált minta milyen információt szerzett a ténylegesen vizsgált modellből, és melyek azok a jellemzők, amelyeket csak a regularizációs prior erőltetett rá.

2.3.1. Optimalizálás a rejtett térben

Sok korábbi AM kutatás az optimalizálást közvetlenül a magas dimenziós képtérben végezte, ahol a pixelenkénti változások gyakran korrelálatlanok, összefüggő struktúrával nem rendelkező vizualizációkat eredményezve. Ehelyett Nguyen és mtsai. [22] javasolják, hogy egy generátor hálózat alacsony dimenziós rejtett terében optimalizáljunk, amit Deep Generator Network Activation Maximization (DGN-AM)-nek neveznek. Egy képgenerátor hálózatot tanítanak be, amely egy tömörített kódot vesz fel és olyan szintetikus képet állít elő, amely a lehető legközelebb áll a valódi képekhez az adathalmazból. Egy adott neuronhoz tartozó AM kép előállításához a szerzők a generátor bemeneti rejtett terében optimalizálnak úgy, hogy az egy olyan képet adjon ki, amely aktiválja a vizsgált egységet. A DGN-AM csak azokra a képekre korlátozza a keresést, amelyek illeszkednek a prior eloszlásba, és ösztönzi a képfrissítéseket, hogy koherensebbek és korreláltabbak legyenek a pixelenkénti változásokhoz képest (ahol minden pixelt függetlenül módosítanak). G generátor terében keresünk, hogy megtaláljunk egy $k \in \mathbb{R}^h$ kódot, amelyre a kép $G(k)$ maximalizálja a neurális aktivációt. Az AM optimalizációja ekkor az alábbi:

$$k^* = \arg\max_k \left(a(G(k)) - R(G(k))\right)$$

Azaz, lépéseket teszünk a rejtett térben az alábbi frissítési szabály szerint:

$$k(t+1) = k(t) + R_{\text{grad}}\left(\varepsilon_1 \frac{\partial a\left(R_{\text{pre}}(G(k(t)))\right)}{\partial k(t)}\right) - \varepsilon_2 \frac{\partial R_{\text{obj}}(G(k(t)))}{\partial k(t)}$$

Az általános formalizációtól való különbség az, hogy R_{pre} regularizációs eljárás ebben az esetben a G generátor hálózat által megvalósított transzformációval kezdődik. Megjegyzendő, hogy itt a regularizáció a rejtett kódot bünteti, nem pedig direktben a képet.

Optimalizálás a mély generátor hálózat rejtett terében a tanult prior alapú célfüggény-regularizációhoz hasonlóan nagyot javít a képminőségen. Azonban annak a hátrányai is jellemzik, ugyanis ez a módszer is egy másodlagos fekete-doboz modell bevezetését igényli. Ezentúl a DGN-AM által szintetizált képek korlátozott változatosságúak, gyakorlatban pedig hasonlóak a valódi top-9 validációs képekhez, amelyek legmagasabbra aktiválják az adott egységet [26].

2.4. Fehér zaj (spike triggered) analízis

A spike triggered analysis algoritmus [19] egy másik olyan módszer, amely segít feltárni a neurális hálózatok egységeinek a vizuális jellemzőkre való érzékenységét és a receptív mezejük térbeli szerkezetét. A receptív mezők közelítésének a vizualizációjához általában a spike triggered average (STA) technikát használják, amely összegyűjti az egység által kiváltott aktivációkhoz tartozó bemeneti mintákat és az aktiváció szerint súlyozottan átlagolja azokat, így becsülve az aktivációk kiváltásához vezető optimális stimulust. Legyen $s(t)$ stimulus vektor, amely a t-edik mintavétele egy generáló zajeloszlásnak, és legyen $a(t)$ a minta által kiváltott aktiváció. Feltételezzük, hogy a stimulusoknak nulla várható értékűek (azaz $E[s(t)] = 0$). Ha nem így van, akkor nullára transzformálhatjuk őket az átlagos stimulus kivonásával. Az STA ekkor a következő:

$$\text{STA} = \frac{1}{N} \sum_t a(t)s(t)$$

ahol N a stimulusok összes száma. Az STA módszer csak akkor ad torzítatlan becslést a lineáris receptív mezőről, ha a stimulus eloszlása gömbszimmetrikus (pl. Gauss-fehér zaj). Gyakorlatban tehát lineárisan tudjuk közelíteni a hálózatunk egységeinek receptív mezőjét nagy mennyiségű fehér-zaj mintának a kiváltott aktivációk szerinti súlyozott átlagolásával [3].

2.5. Generatív modellek

A generatív modellek célja, hogy megtanulják a valós adatok eloszlását és olyan mintákat generáljanak, amelyek hasonlítanak az eredetihez, de nem másolják azokat. Példák a generatív modell architektúrákra az energia-alapú modellek, a variációs autóenkóderek (VAE), a generatív versengő hálózatok (GAN), autoregresszív modellek, normalizáló folyamat (normalizing flows) modellek és számos hibrid megközelítés.

Egy $D = x(1), ..., x(N)$ adathalmaz alapján valósághű adatpontok $x \in \mathbb{R}^d$ generálása során a generatív modellek általában feltételezik, hogy létezik egy háttéreloszlás μ_{gt}, amely alacsony dimenziós sokaságon $\psi \subset \mathbb{R}^d$ $k < d$ dimenzióval koordinátázva abszolút folytonos a ψ-n definiált Hausdorff-mérték szerint $p_{\text{gt}}(x)$ sűrűséggel [1]. Ezzel a feltételezéssel élve átírhatjuk:

$$p_{\text{gt}}(x) = \int_{\mathbb{R}^k} p_{\text{gt}}(x, z)\, dz = \int_{\mathbb{R}^k} p_{\text{gt}}(x|z)\, p(z)\, dz = \mathbb{E}_{p(z)}[\, p_{\text{gt}}(x|z)]$$

Ahol $z \in \mathbb{R}^k$ az x-hez társított rejtett változó, amely egyszerű eloszlással $p(z)$ rendelkezik (amelyet prior eloszlásnak neveznek). A generatív modellek mögötti ötlet az, hogy ha megtanulhatunk egy jó közelítést a $p_{\text{gt}}(x|z)$-re az adatokból, akkor ezt a közelítést felhasználhatjuk új minták generálására mintavételezéssel, azaz mintavételezzük $z \sim p(z)$, majd generáljuk $x \sim p_{\text{gt}}(x|z)$.

Gyakori, hogy egy $\mathcal{P}(\theta) = \{p_\theta(x|z)|\theta \in \mathbb{R}^s\}$ paraméteres valószínűségi családot határozunk meg egy neurális hálózattal, és megtaláljuk θ^*-ot, amelyre teljesül, hogy

$$\theta^* = \arg\max_\theta \mathbb{E}_D \left[\log p_\theta(x)\right] = \arg\max_\theta \mathbb{E}_D \left[\log \int_{\mathbb{R}^k} (p_\theta(x|z)p(z)dz)\right]$$

azaz a paramétereket Maximum Likelihood Becslés alapján tanuljuk. A feladatunkra nézve a generatív neurális hálózatok fontos tulajdonsága, hogy lehetővé teszik $p_\theta(x) \sim\ p_{gt}(x)$ kiszámítását, vagy legalábbis ennek az értéknek egy közelítését. Ki tudjuk tehát számítani egy adott mintának a valószínűségét a tanítóminták alapján konstruált modellezett eloszlás szerint.

2.5.1. Variational Autoencoder

Sajnos a Maximum Likelihood Becslés általában számításilag megvalósíthatatlan (intractable). A VAE ötlete [14], hogy egy másik valószínűségi eloszlást határoz meg ($q_\phi(z|x)$ jelöli), amely kódoló eloszlásként ismert, és leírja egy adatpont $x \in \chi$ és a rejtett változó $z \in \mathbb{R}^k$ közötti kapcsolatot úgy, hogy optimalizálja ϕ-t és θ-t az alábbiak szerint:

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathbb{E}_D \left[D_{KL} \left(q_\phi(z|x) \parallel p_\theta(z|x) \right) \right]$$

Vegyük figyelembe, hogy:

$$D_{KL} \left(q_\phi(z|x) \parallel p_\theta(z|x) \right) = \mathbb{E}_{q_\phi(z|x)} \left[\log q_\phi(z|x) - \log p_\theta(x|z) - \log p_\theta(z) + \log p_\theta(x) \right]$$

Ezért:

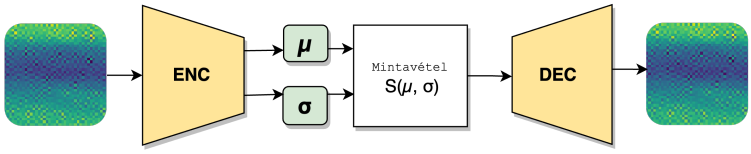
$$\begin{aligned} &\mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z) - D_{KL} \left(q_\phi(z|x) \parallel p(z) \right) \right] && (1) \\ &= \log p_\theta(x) - D_{KL} \left(q_\phi(z|x) \parallel p_\theta(z|x) \right) && (2) \\ &\leq \log p_\theta(x) && (3) \end{aligned}$$

Mivel $D_{KL}(q_\phi(z|x) \parallel p_\theta(z|x)) \geq 0$, ezért (1) egy alsó korlát a $p_\theta(x)$ logvalószínűségére. Ezt azt az alsó korlátot ELBO-nak (Evidence Lower Bound) nevezik. Mivel az ELBO kezelhetőbb, mint az ML becslés, ezért ezt használjuk költségfüggvényként a neurális hálózat tanításához (optimalizálja mind a θ, mind a ϕ paramétereket). Normál eloszlást feltételezve ebben mindegyik tag zárt formában kiszámítható [1].

$$L_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - D_{KL}[(q_\phi(z|x) \parallel p(z))]$$

$$L_{\theta,\phi} = \mathbb{E}_D \left[L_{\theta,\phi}(x) \right]$$

A VAE tehát gyakorlatban egy $q_\phi(z|x)$-t modellező kódoló neurális hálózatból (recognition model), egy $p_\theta(x|z)$-et modellező dekódoló neurális hálózatból (generator model) és egy $p(z)$-t modellező prior eloszlásból áll (2. ábra). Tanítás során ezen eloszlásokból mintavételezve számítjuk ki a költséget. A generatív mintavételezéshez pedig elfelejtjük a kódolót, és csak a dekódert használjuk. Ilyenkor a rejtett változókat a $p(z)$ prior eloszlás szerint mintavételezzük.



2. ábra. A Variational Autoencoder (VAE) architektúra felépítése. A bemeneti képet ENC egy paraméteres $S(\mu, \sigma)$ eloszlásra projektálja. Ebből mintavételezve DEC rekonstruálni próbálja az eredeti képet.

2.5.2. Reparameterization trick

A modellben való mintavételezés során azonban mintavételezési zaj jelenik meg a tanításhoz használt gradiensekben. Problémaként merül fel tehát, hogy hogyan lehet integrálni a mintavételezést a backpropagation algoritmussal. Erre a megoldást az úgynevezett "reparameterization trick" jelentheti [14]. Itt a mintát egy standard eloszlásból vesszük (a backpropagation áramlásán kívül), majd ezt transzformáljuk $\mu_\phi(x)$ és $\sigma_\phi(x)$-el, melyeken keresztül a hiba terjeszthetővé válik.

2.5.3. Top-Down VAE (TDVAE)

A VAE modellek speciális típusai a hierarchikus VAE modellek. Ezek a látens változók több rétegbe szervezésével komplexebb priorok építését teszik lehetővé. Formálisan [2]:

$$p_\theta(z) = p_\theta(z_0)p_\theta(z_1|z_0)\dots p_\theta(z_N|z<N)$$
$$q_\phi(z|x) = q_\phi(z_0|x)q_\phi(z_1|z_0,x)\dots q_\phi(z_N|z<N)$$

Egy kétrétegű hierarchikus modell esetében a látens változók poszteriorját kétféleképpen faktorizálhatjuk. Ez bottom-up módon így néz ki:

$$p(z_1,z_2|x) = p(z_2|x,z_1)\cdot p(z_1|x)$$

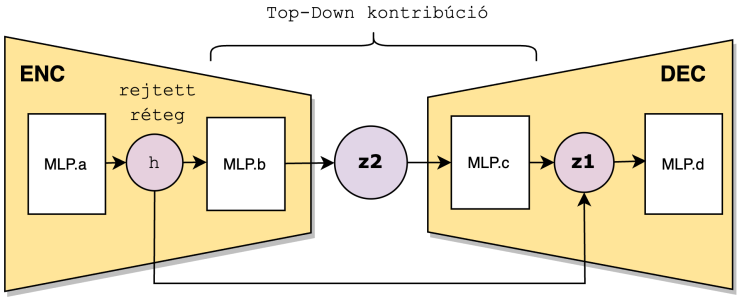
Azonban faktorizálható top-down módon is a következőképpen:

$$p(z_1,z_2|x) = p(z_1|x,z_2)\cdot p(z_2|x)$$

A Top-Down VAE [4] a látens változók poszteriorjának ezen faktorizációját modellezi. Ebből kiszámolható a modell tanítási költségfüggvényeként használt ELBO, ami a következőképpen néz ki:

$$\begin{aligned}\text{ELBO}(x,\theta,\Phi) =& \mathbb{E}q_\Phi(z_2|x)q_\Phi(z_1|x,z_2)[\log p_\theta(x|z_1)] \\ &- \mathbb{E}q_\Phi(z_2|x)\left[\text{KL}\left[q_\Phi(z_1|x,z_2)||p_\theta(z_1|z_2)\right]\right] \\ &- \text{KL}\left[q_\Phi(z_2|x)||p_\theta(z_2)\right].\end{aligned}$$

A TDVAE architektúrában megjelenő felülről lefelé irányuló hatások a felismerési modellben (recognition model) gazdagabb reprezentációt eredményeznek, és képesek reprodukálni néhány kulcsfontosságú tulajdonságát a V1 és V2 látókéregben kialakuló reprezentációknak [4]. Ez úgy jelenik meg, hogy bizonyos információk, amelyek nincsenek jelen az átlagos aktivációkban, lineárisan dekódolhatóvá válnak a poszterior korrelációkból. A szakirodalomban a poszterior korrelációkat összefüggésbe hozták a neuronális zajkorrelációval (NC) [9][30][5]. Ezt az eredményt összekapcsolták a korábbi megfigyelésekkel a majom elektrofiziológiában, ahol az NC stimulus-specifikussága megváltozott, amikor a stimulus statisztikáit manipulálták. Vizsgálták a felülről lefelé irányuló hozzájárulásokat a kontextuális hatásokhoz is. Hasonlóan a V1 neuronok aktivitásához [16], azt találták, hogy illuzórikus élek jelenhetnek meg a látens válaszokban, és ez fokozódik a felülről lefelé irányuló kapcsolattal. Hasonló hozzájárulást azonosítottak a kép befestésében, ahol a maszkolt képeket egészítették ki, a felülről lefelé irányuló kapcsolatok által szolgáltatott információkra támaszkodva [4].



3. ábra. A Top-Down VAE (TDVAE) architektúra felépítése. A bemenetet h rejtett rétegen keresztül egy $z2$ eloszlásba vetítjük, majd az itt eltárolt információ segítségével (Top-Down kontribúció) számítjuk ki $z1$ elszoszlást. $z1$-ből lineárisan dekódolható a rekonstrukció.

3. Kísérletek

3.1. Vizsgált modell

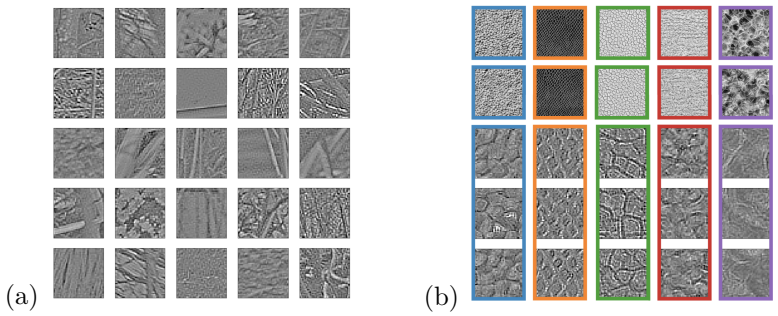
A kísérletek során egy kétrétegű Top-Down VAE modell [4] egységeit vizsgálom. A modellt egy x bemenetből $\hat{x}$ rekonstrukciójára tanították z_1 és z_2 látens rétegeken keresztül. A modell felépítését a 3. ábra mutatja. Az architektúra tehát 5 eloszlásfüggvényt modellez, mindegyiket egy Multi Layer Perceptron (MLP) hálózattal (1. táblázat).

Az első neurális hálózat (MLP.a) a pixelteret rendeli egy h_x réteghez, amit $q_\Phi(z_2|x)$ és $q_\Phi(z_1|x, z_2)$ megosztanak a számításaikhoz. h_x-ból MLP.b kiszámítja $q_\Phi(z_2|x)$ normál eloszlás várható értékét és varianciáját. A harmadik (MLP.c) z_2-t alakítja át h_z réteggé. Majd h_x és h_{z_2} összefűzésével összegzzük az x-beli és a z_2-beli információt és erre alkalmazzuk a negyedik neurális hálózatot (MLP.d), hogy kiszámítsuk $q_\Phi(z_1|x, z_2)$ eloszlás várható értékét és varianciáját. A modellben z_1 réteg túltelített ($\dim(z_1) > \dim(x)$) rétegként van megvalósítva, amelynek egy sparse (Laplace) prior eloszlása van, és amely lineáris generatív kapcsolatban van a megfigyelésekkel: $p_\theta(x|z_1) = \mathcal{N}(x; Az_1, I)$, ahol A egy lineáris transzformáció mátrixa, I pedig egységmátrix. Az ötödik (MLP.e) $p_\theta(z_1|z_2)$ prior eloszlást számítja ki. A modellben lévő MLP hálózatokban az aktivációk Softplus nemlinearítások kimenetei.

MLP.a	MLP.b	MLP.c	MLP.d	MLP.e
(2000)	(1000, 500, 250)	(250, 500, 1000, 2000)	(2000)	(2000)

1. táblázat. A TDVAE által használt MLP neurális hálózatok teljesítményrétegeinek és kimeneti rétegeinek (utolsó a felsorolásokban) dimenziószámai. Forrás: [4]

A vizsgált modellt 40x40 méretű egycsatornás textúrák rekonstrukciójára tanították természetes képeken (4. ábra). A természetes képek egy jellegzetes lineáris szerkezetet mutatnak [29]. Lineáris modell által nem leírható magasabb rendű függőségeket [34] és nemlineáris jellemzőket is azonosítottak a természetes képekben [12]. Ez biztosítja, hogy ezek kitűnő tesztágyat nyújtsanak a hierarchikus generatív modellekben kialakuló reprezentációk vizsgálatához.

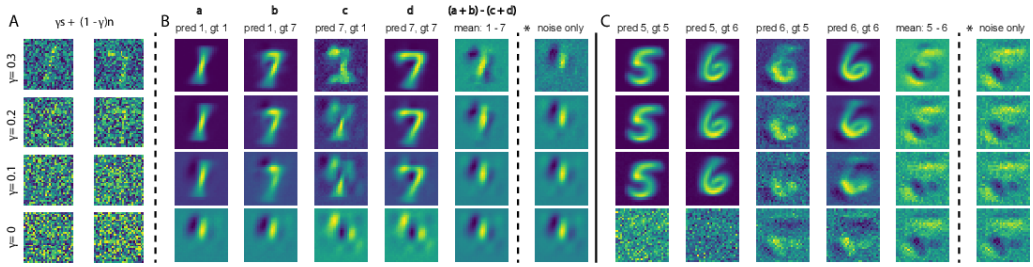


4. ábra. (a) Példa 40 × 40-es fehérített természetes textúra képekre a tanító adathalmazból. (b) Első sor: kivágások az öt textúracsalád generálási kiindulóképeiből. Második sor: kivágások a [34] által szintetizált textúrákból. Alsó sorok: 40 × 40-es fehérített textúrarészletek, amelyeket a szintetizált textúraképekből vágtak ki. Forrás: [4]

A dolgozat során z_2 réteg tanult reprezentációit fogom vizsgálni. A z_2 dimenziók aktív és összeomlott csoportokba oszlanak. Utóbbiakat a poszteriális z_2 átlagok kis szórásai és a posteriális z_2 szórások egységhez közeli átlagai jellemezik az egyes textúracsaládokon belül. A tanító adatok mellett 5 külön textúra-családba tartozó szintetikus adathalmazt is generáltak [4] Portilla és Simoncelli [34] optimalizáló algoritmusa alapján. A z_2-ben lévő aktiváció átlagokra a textúra családokat klasszifikáló modellt tanítottak be, aminek segítségével észrevették, hogy vannak textúra-érzékeny z_2 dimenziók. A dolgozatban 5 db textúra-érzékeny aktív z_2 dimenzió és 5 db nem textúra érzékeny aktív z_2 dimenzió vizsgálatán keresztül kísérletezem a preferált bemenet előállítására alkalmas módszerekkel.

3.2. Fehér-zaj analízis

A Spike Triggered Avarage (SPA) módszer alkalmazásával vizsgálom az egységek receptív mezőinek lineáris közelítését. A folyamathoz először is szintetizálok nagy mennyiségű nulla autokorrelációs értékkel rendelkező fehér zaj bemenetet a vizsgált modell bemeneti terében. Ezekhez hozzárendelek egy aktivációs értéket az alapján, hogy a modell vizsgált neuronjaiból milyen aktivációt váltottak ki. Végül ezen aktivációs értékekkel súlyozva átlagolom a fehér-zaj mintákat.



5. ábra. Fehér-zaj analízissel előállított, az MNIST adathalmazon osztályzó neurális hálózatban az egyes oszályokhoz tartozó preferált bemenetek. Forrás: [3]

3.3. Generatív modell képszintézis

Generatív modellek a preferált bemeneteinek megtalálására kihasználhatjuk a modell generatív létét is a szintetizáláshoz. Ilyenkor mintavételezünk a modell által megtanult prior eloszlásokból. Legyen z_2* egy skálázott one-hot vektor, ahol a megvizsgálni kívánt egység értéke egy magas pozitív érték, az összes többié pedig nulla. Ezzel kiszámíthatjuk a $p_\theta(z_1|z_2*)$ prior eloszlást, majd ebből mintavételezve, vagy a várható értékét véve, megkapjuk z_1-et. Ennek a segítségével pedig $p_\theta(x|z_1)$ megtanult eloszlásból mintavételezhetünk, így szintetizálva egy x mintát.

Ez a módszer nem ugyanarra a kérdésre ad választ, mint az aktiváció maximalizáció, ugyanis abban az esetben nem tettünk olyan megkötést, hogy a vizsgálton kívül minden másik dimenzió nulla (vagy egyáltalán alacsony) értékű legyen z_2-ben. Ennek ellenére a két feladat közti hasonlóságok indítékot adnak a köztük lévő kapcsolat vizsgálatára.

3.4. Aktiváció Maximalizálás

3.4.1. Célfüggvény

Az AM feladata általános probabilisztikus megfogalmazásban $p(x|a, D)$ megtalálása úgy, hogy $x \sim p(x|a, D)$ egy ember által értelmezhető reprezentációja legyen az egység által megtanult jellemzőknek. Egy D adathalmazon x bemenetre és a aktivációra értelmezve:

$$p(x|a, D) = \frac{p(x, a, D)}{p(a, D)} = \frac{p(a|x) \cdot p(x|D) \cdot p(D)}{p(a, D)} = \frac{p(a|x) \cdot p(x|D) \cdot p(D)}{p(a|D) \cdot p(D)} \propto p(a|x) \cdot p(x|D)$$

Ahol $p(a|x)$ a vizsgált modellünk egy egységének aktiválódása egy x bemenetre, $p(x|D)$ pedig egy regularizációs tényező. Kézzel tervezett priorok esetében feltételezzük, hogy ez a regularizációs tényező független D adathalmaztól, tehát $p(x|D) = p(x)$. Ez a megengedés azonban nagyban megnehezíti az értelmezhető minták szintetizálásának a folyamatát. Tanult prior esetében $p(x|D)$ modellezése az eddigi munkákban egy másodlagos neurális hálózat bevezetését jelentette. Generatív modellek vizsgálatakor azonban ez a valószínűség, vagy ennek egy közelítése kiszámítható, és eléri a bemenet gradiensét.

A regularizált AM általános leírása szerint x képen egy véletlen inicializált állapotból kiindulva lépésenkénti optimalizációt végzünk a következő frissítési szabállyal:

$$x(t+1) = R_{\text{post}}(x(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(R_{\text{pre}}(x(t)))}{\partial x(t)}) - \epsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)})$$

Amivel $R_{\text{obj}}(x)$ egy általánosítása $p(x|D)$ valószínűségnek felírható a következőképpen:

$$x(t+1) = R_{\text{post}}(x(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(R_{\text{pre}}(x(t)))}{\partial x(t)}) + \epsilon_2 \frac{\partial p(x|D)}{\partial x(t)}))$$

Gyakorlatban ez a valószínűség sokszor nem számítható integrált tartalmaz. Ebben az esetben közelítést adhatunk rá $p_{\text{közelítés}}(x) \approx p_{\text{model}}(x)$ formájában. Számításilag megoldhatatlan integrál esetén a valószínűségi tag közelítésére kézenfekvő megoldás a generatív modell tanítási költségfüggvénye. Ezt az állítást a generatív modellek elveire alapozom, ugyanis ezek kimondják, hogy ezen modellek fő feladata a tanítási adathalmaz mögött elterülő $p(x)$ háttéreloszlás reprezentálása minél jobb $p_\theta(x)$ közelítéssel. Míg ezen modellek tanítása során θ paramétereket optimalizáljuk, hogy $p_\theta(x) \approx p(x)$, most rögzítjük θ paramétereket, és x generált bemenetet szeretnénk $p_\theta(x)$, a modell által megtanult háttéreloszlás irányába terelni. Ez arra fogja ösztönözni az optimalizációt, hogy az általa generált minták közel essenek a generatív modell által modellezni kívánt háttéreloszláshoz. A valószínűség kiszámításához ez esetben nincs szükség egy másodlagos neurális hálózat betanítására. p_{reg}, a közelítési hibát korrigáló tényezővel az alábbi alakra jutunk:

$$p(x|D) = p_{\text{közelítés}}(x) \cdot p_{\text{reg}}(x)$$

VAE modellek esetében az Evidence Lower Bound (ELBO) alsó becslést ad $p(x)$-re. Feltételezzük, hogy a modell jól megtanulta a modellezni kívánt háttéreloszlást, tehát $p_\theta(x) \approx p(x)$. Így az ELBO közelítő becslést ad $p_\theta(x)$-re is. Mi pedig az optimalizációs feladatunkra pont egy ilyen közelítést kerestünk.

$$p(x|D) = p_{\text{elbo}}(x) \cdot p_{\text{reg}}(x)$$

A TDVAE modell esetében tehát így néz ki a frissítési szabály:

$$x(t+1) = R_{\text{post}}(x(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(R_{\text{pre}}(x(t)))}{\partial x(t)}) + \epsilon_2 \frac{\partial ELBO(\text{TDVAE}, \theta, x(t))}{\partial x(t)})$$

3.4.2. Optimalizáló

Az újítások a neurális hálózatok paramétereinek az optimalizációjában olyan fejlesztéseket hoztak a hagyományos Stochaistic Gradient Descent (SGD) algoritmushoz, amelyek sok esetben felgyorsítják és akár stabilizálják a tanulási folyamatot. A mi feladatunk azonban a maximalizálás, amelyre formálisan a Stochaistic Gradient Ascent algoritmust használjuk. Gyakorlatban azonban egyszerűen a költség ellentettjét véve minimalizálási problémára fordíthatjuk, így lehetőséget nyerve a preferált bemenetek generálásához eddig nem vizsgált "state-of-the-art" optimalizációs algoritmusok alkalmazására. Az egyszerű Stochastic Gradient Descent (**SGD**) algoritmus egyike a leggyakrabban használt módszereknek a modellparaméterek frissítésére, azonban ez az algoritmus kiemelten érzékeny a frissített paraméterek szerinti érzékenység kondícionáltságára, ami a konvergencia lassulásához vezethet [32]. A [41]-ben használt **MEI** optimalizációs algoritmus egy olyan továbbfejlesztett változat, amely kompenzálja ezt a problémát. Az egyenletes frissítések elérése érdekében a gradiens méretével normalizál (adaptív bátorsági tényezőként):

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{E[g]_t}{\|E[g]_t\|_1} \tag{4}$$

Az **RMSprop** (Root Mean Square Propagation) algoritmus egy olyan, szélesebb körben alkalmazott, másodrendű momentumos algoritmus, amelyet szintén a paraméterenkénti eltérő érzékenység problémája motivált. Az RMSprop RMS átlaggal becsli a súlymódosítás irányonkénti nagyságát, mellyel normálva, egyfajta irányonkénti adaptív bátorsági tényezőként, kompenzálja a paraméterek szerinti eltérő érzékenységeket. Tehát g gradiens esetén a következő frissítést hajtjuk végre:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$
$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{E[g^2]_t + \epsilon}} \odot g_t$$

Az **Adam** (Adaptive Moment Estimation) [13] algoritmus, az RMSprop és a Polyak Momentum [33] algoritmusok kombinációjával született meg. Ezáltal kompenzálni tudja az egyes optimaliálandó paraméterek szerinti eltérő érzékenységet, továbbá stabilizálni tudja a paraméterenkénti oszcillációt is a Polyak Momentum használatával. Az optimalizálásra tehát négy különböző gradiens alapú paraméter optimalizálót vizsgálok. Utánajárok, hogy különböző értékű lépésméretek és lépésméret-ütemzések milyen hatással vannak az optimalizációra, továbbá vizsgálom, hogy ezek az értékek milyen kapcsolatban állnak az értelmezhető eredmény eléréséhez szükséges iterációszámmal.

3.4.3. AM típusai

Dolgozatomban a hagyományos pixeltér-beli (vagy látens kódtér-beli) optimalizáción túl javaslatot teszek két új paraméterezési modellre.

- (1) Feltételezem, hogy azon minták eloszlása, amelyek a legjobban aktiválják a vizsgált egységet, egyszerű (normál vagy Laplace). A pixeltér-beli optimalizáció helyett ennek az **eloszlásnak a paraméterein** végzem az optimalizációt.

- (2) Az optimalizációt egy bemenet és egy tanulható **transzformáció (pl. konvolúció, neurális hálózat) paraméterein** végzem. A mintákat a bemenetre a transzoformáció elvégzésével generálom. Majd egy hibrid megközelítésben, hasonlóan az (1)-hez, feltételezek egy egyszerű eloszlást a preferált bemenetek felett, és tanulható transzformációt végzek el ezen eloszlás paraméterein.

3.5. Pixeltér alapú AM

3.5.1. Módszer

A pixeltérben való optimalizáció a AM hagyományos változtata, tehát a folyamatot $x \in \mathbb{R}^{H \times W \times C}$ paraméteren végezzük (6. ábra).

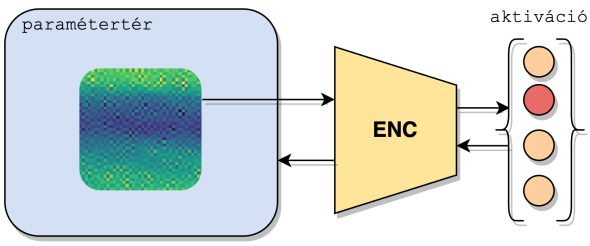
3.5.2. Regularizáció

A szakirodalomban bevezetett regularizációs technikákat hasonlítom össze.

- Az optimalizációs lépés előtti $R_{\text{pre}} : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{H \times W \times C}$ transzformációra a bemeneti minta jitter-elésével, skálázásával és normalizálásával kísérletezem. Ezek a műveletek nem szakítják meg a számítási gráfot, így backpropagation során az aktiváció el tudja érni a bemenet gradiensét.
- Az optimalizációs lépés utáni $R_{\text{post}} : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{H \times W \times C}$ regularizációs transzformációra megnézem milyen eredményeket produkálnak az iterációnként elvégzett különböző zajtalanító műveletek. A vizsgált zajtalanító eljárások közé tartozik (1) a képen Gauss-elmosás elvégzése aluláteresztő szűrővel; (2) a kép zajtalanítása bilaterális szűrővel az élek megtartása érdekében; valamint (3) Total Variation (TV) alapú optimalizáció a képen. Ezentúl kísérletezem az optimalizációs lépés utáni kép normalizálásával is.
- A gradiensképen végzett $R_{\text{grad}} : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{H \times W \times C}$ transzformációra is a Gauss elmosással próbálkozom.
- A célfüggvényben a bemeneti képre kiszámolt $R_{\text{obj}} : \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{1}$ regularizációs tag kontribúcióját is vizsgálom az eredményekre. Gyakorlatban az ELBO egy skálázott értékét vagy a képre kiszámolt totális varianciát adjuk hozzá az optimalizációs költséghez.

3.5.3. Változatosság

Változatosság eléréséhez több független mintát optimalizálok, és az optimalizációs célfüggvényébe bevezetek egy változatossági tényezőt [27]. Ez arra fogja ösztönözni a generált mintákat, hogy legyenek minél távolabb egymástól. Ennek a távolságnak a meghatározására több metrikát is kipróbálok, ezek közé tartozik a koszinusz hasonlóság, a minták közti korreláció, valamint a köztük lévő euklideszi távolság [41].



6. ábra. A Pixeltér alapú AM esetén az optimalizációt egy kép pixelein végezzük. Kiszámítjuk a képre vett aktivációt, majd eszerint változtatjuk a pixelek értékeit.

3.6. Eloszlás alapú AM

3.6.1. Módszer

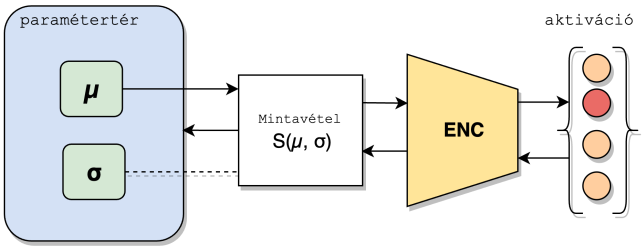
A pixel-téren való optimalizáció fő problémája, hogy az általa generált képeken sok magas-frekvenciás zaj jelenik meg. A probléma orvoslására kísérletezem az eloszlás alapú AM optimalizációval. Feltételezzük, hogy a magas aktivációt kiváltó bemenetek egy egyszerű eloszlásra (normál vagy Laplace) illeszkednek, és egy minta megtalálása helyett ezt az eloszlást szeretnénk megtalálni (7. ábra).

Legyen $x_1, x_2 \in \mathbb{R}^{H \times W \times C}$ egy egyszerű eloszlás (normál vagy Laplace) várható értéke és szórása. A probléma az előzőekhez hasonló módon leírható: keressük azon x_1, x_2 paramétereket (egy eloszlás parametrikus leírása), ami a legnagyobb aktivitást váltja ki a vizsgált egységünkből. Azt is vizsgálom, hogy hogyan változik az eredmények minősége, ha feltételezzük, hogy az eloszlás kovariancia mátrixa diagonális. Itt mindössze a várható értéket optimalizáljuk, és egy fix szórásértékkel dolgozunk. A módszerben az optimalizációs lépés a következőképpen formalizálható:

$$x_1(t+1) = x_1(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(\mathrm{R}_{\text{pre}}(x(t)))}{\partial x_1(t)}) - \epsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t}),$$

$$x_2(t+1) = x_2(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(\mathrm{R}_{\text{pre}}(x(t)))}{\partial x_2(t)}) - \epsilon_2 \frac{\partial R_{\text{obj}}(x(t))}{\partial x(t)}) \text{ vagy } x_2(t),$$

ahol $x(t)$ egy mintavételezett batch $S(x_1, x_2)$ kétparaméteres egyszerű eloszlásból. Vegyük észre, hogy nem jelenik meg a R_{post} regularizációs tag, mivel ez a gyakorlatban képekre vett regularizációt jelent, így nem feltétlenül alkalmazható intutív módon eloszlások paramétereinek a regularizációjára. A mintavételezéshez a "reparameterization trick"-et alkalmazzuk, ami a gyakorlatban a szórás által skálázott véletlen zaj hozzáadását jelenti a várható értékéhez. Ez lehetővé teszi, hogy egyszerre több bemenetet is mintavételezzünk, majd ezeket egy batch-ben adjuk oda a hálózatnak, az optimalizációs lépést egy egész batch után megtéve. Ennek több előnye is lehet: (1) A batch-ek növelhetik a tanulás stabilitását és konvergenciáját, mivel a súlyfrissítések átlagolódnak a batch-en belüli adatokon, és így kevésbé érzékenyek az egyedi zajra vagy kiugró értékekre. (2) A batch-ek lehetővé teszik, hogy a számítógép több adatot dolgozzon fel egyszerre és kihasználja a párhuzamos számítás előnyeit. Ez csökkenti a memóriaigényt és növeli a hatékonyságot. Kísérletezem a pixeltér esetében ismertetett regularizációs technikákkal, illetve vizsgálom a megtanult eloszlásban a várható érték és a variancia szerepét a minták minősége és változatossága szempontjából.

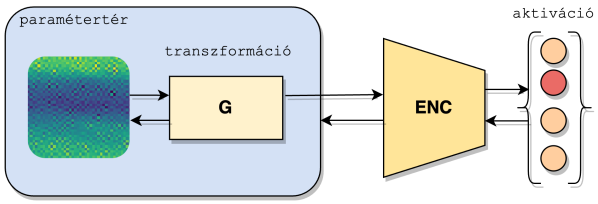


7. ábra. Az Eloszlás alapú AM esetében μ és σ paramétereken végezzük az optimalizációt. Mintavételezünk $S(\mu, \sigma)$ eloszlásból és mintavételekre kiszámított aktivációk alapján változtatjuk μ-t és σ-t.

3.7. Transzformáció alapú AM

3.7.1. Módszer

Mind a pixeltéren lévő, mind az eloszláson alapuló AM módszernek vannak korlátozásai. A vizsgált textúrákra érzékeny egységek aktivációjára való optimalizálás gyakran nehezen értelmezhető, inkonzisztens és zajos eredményeket produkál. Koherens minták generálásához kézi regularizációra van szükség. Ezek a regularizációs technikák azonban egyszerű heurisztikákon alapulnak. Újabb paraméterek bevezetésével azonban tanulhatóvá tehetjük a regularizációs folyamatot is. Erre a gondolatra alapozva új megoldást javaslok az AM problémájára.

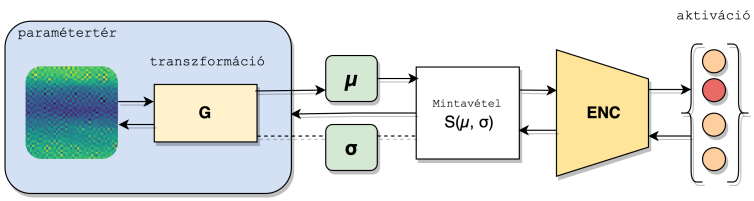


8. ábra. A Transzformáció alapú AM egy bemeneti jelet és egy regularizációs transzformáció paramétereit optimalizálja. A mintákat a bemeneti jel transzformáltjaként kapjuk.

Legyen $i \in \mathbb{R}^d$ választható d dimenziójú bemeneti jel és egy $G : \mathbb{R}^d \to \mathbb{R}^{H \times W \times C}$ transzformáció. A paramétertér ebben az esetben $x = \{i, \theta\}$, ahol θ jelöli a G transzformáció paramétereit. A mintavételezés úgy történik, hogy i bemeneten elvégezzük G transzformációt, és ennek eredményeként kapjuk a szintetizált mintát (8. ábra). Az optimalizáció frissítési lépése a következőképpen alakul:

$$x(t+1) = x(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(\mathrm{R}_{\text{pre}}(G(i)))}{\partial x(t)} - \epsilon_2 \frac{\partial R_{\text{obj}}(G(i))}{\partial x(t)})$$

Vizsgálom az eloszlás alapú és a transzformáció alapú megközelítések egy **hibrid** változatát is. A vizsgált egységből magas aktivációt kiváltó bemenetek mögött itt is egy egyszerű eloszlást feltételezek. Azonban most nem egyenesen az eloszlás paraméterein végezem az optimalizációt, hanem azokat egy $G(i)$ transzformációval állítom elő. Itt $G(i)$ transzformációnak az eredménye tehát egy egyszerű eloszlás paramétereit adja. Ebből az eloszlásból az előzőhöz hasonlóan történik a mintavételezés. (9. ábra)



9. ábra. A Transzformáció + eloszlás alapú AM: a képre elvégzett transzformáció egy eloszlást paraméterez. Ebből mintavételezve kapjuk a bemeneteket.

A transzformációhoz gyakorlatban konvolúciós műveletet, majd ezt bővítve konvolúciós hálózatot (CNN) használok. Az ilyen modellek nagy sikereket értek el az utóbbi évtizedben mind a képfelismerés, mind a képszintézis területén. A CNN-ek használata a textúrák szintetizálásához különösen indokolt. A textúrák generatív definíciója szerint egy textúra egy olyan kép, amelynek minden részlete ugyanolyan eloszlásból származik. A CNN-k ezt a definíciót használják, amikor a textúrákat a hálózat különböző rétegeiben lévő jellemzőtérképek korrelációival reprezentálják [7]. Konvolúciós műveletek esetén a megtanult transzformációk egy szűrésként viselkednek: modulálják a képek spektrumainak amplitudóit, ezáltal elnyomnak olyan frekvenciájú komponenseket, melyek inkoherens zaj alakzatok rekonstrukciójáért felelnének. Ez hasonlít egy alacsonyabb dimmenziós sokaságba kényszerítésre, mely segít elkerülni a szűrések kiemeneti képeinek összeomlását, illetve segít értelmes, koherens mintázatú eredményt elérni.

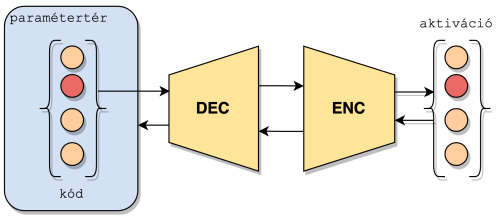
3.7.2. Megtanult generátor modell alapú AM

A szakirodalomban megjelenő rejtett térben való optimalizálásra alapuló módszerek mind egy, az adathalmazon külön betanított G neurális hálózatra épülnek. Megtanítják őket, hogy a modell egy rejtett rétegében lévő kódból az adathalmaz eloszlásába tartozó mintákat szintetizáljon, és az optimalizációt ezen a kódon végzik. Generatív modellek vizsgálatakor azonban ugyanezt elvégezhetjük egy külön modell betanítása nélkül is. Definíciójukból adódóan ezek a modellek alkalmasak képszintézisre, ami általánosítva egy $p(z)$ priorból való mintavételezést, majd $p(x|z)$ priorból való mintavételezést jelent. Ezt a tulajdonságot felhasználhatjuk a preferált bemenetek előállítására is. Egy k kódot keresünk ($\dim(k) = \dim(z)$), amit felhasználva $p(x|z = k)$-ből való mintavételezéshez, egy olyan $x \in \mathbb{R}^{H \times W \times C}$ mintát kapunk, ami maximalizálja a neurális aktivációt. A regularizációs transzformáció DEC tehát ebben az esetben nem tanulható, hanem a vizsgált modellünk priorjából egy k kóddal való mintavételezés (10. ábra).

Azaz, lépéseket teszünk a rejtett térben az alábbi frissítési szabály szerint:

$$k(t+1) = k(t) + R_{\text{grad}}(\epsilon_1 \frac{\partial a(R_{\text{pre}}(DEC(k(t))))}{\partial k(t)}) - \epsilon_2 \frac{\partial R_{\text{obj}}(DEC(k(t)))}{\partial k(t)}$$

ahol k egy látens kód, DEC pedig a vizsgált generatív neurális hálózatból való mintavételezési eljárás.



10. ábra. A Megtanult generátor alapú AM esetében egy rejtett kódon végezzük az optimalizációt. Ezzel a kóddal mintavételezünk a generatív modellből, majd az így kapott minták aktivációját viszgáljuk.

Gyakorlatban a VAE modellek esetén DEC a modell dekóder (generator model) részének levágása, a feladat pedig az ehhez való bemenet dimenzióinak terében k kód keresése. Kísérletezem a k kódra való optimalizációval a TDVAE mind z_1, mind z_2 rétegének terében.

3.8. Validáció

Lefuttattuk az optimalizációt, tehát a megadott mennyiségű iteráción keresztül egy célfüggvény minimalizálásának irányába hangoltuk a paramétereinket. Mivel a neurális hálózatok működése jelentősen eltér az emberi intuícióktól, nehéz olyan képeket előállítani, melyek számunkra interpretálhatóak, és a neuronok érzékenységét is jól kifejezik. Hasonlóan nehéz feladat egy-egy ilyen képnek a validációja, melyre heurisztikus módszereket alkalmazhatunk. Ezeket a validációs módszereket két kategóriába, kvantitatív és kvalitatív osztályokba soroltam, és munkámban kapcsolatot keresek a két megközelítés között.

3.8.1. Kvantitatív validáció

A kvantitaív validációs módszerek olyan metrikákat foglalnak magukba, amelyek számszerű értékeket, és ezen értékek statisztikáit vizsgálják. Dolgozatomban a következő kvantitatív validációs technikákat alkalmaztam:

- a vizsgált egység aktivációjának vizsgálata. Ez az érték becsapós lehet, mivel ennek a maximalizálására optimalizáltunk, mégsem tudhatjuk, hogy akár egy lokális optimumot találtunk-e. Ennek ellenére egyértelműen korrelál az optimalizáció "sikeressége" az aktivációval. Továbbá informatívak lehetnek az oszcilláló/divergáló aktivációk is.

- a vizsgált egység aktivációjának összehasonlítása az azonos rétegben lévő többi egység aktivitásával (pl. a második legnagyobb aktivitású egységével). Ezzel azt vizsgáljuk, hogy tényleg a keresett egység preferált bemenetére találtunk-e egy becslést, vagy az egy másik, esetleg több másik egység megtanult tulajdonságait (is) ábrázolja.

- a bemenet modell-hűségének, tehát $p_{\text{model}}(x)$ értékének, vagy ennek a kiszámítható közelítésének (pl. p_{elbo}(x)) nyomon követése. Ez az érték megválaszolja, hogy a szintetizált bemenet mennyire áll távol a vizsgált modell által megtanult háttéreloszlástól.

Ezeket az metrikákat minden iterációban vizsgálom, majd egy grafikonon vizualizálom, hogy miképpen változott az értékük az optimalizációs folyamat alatt. Az optimalizációt sikeresnek nevezem, ha a folyamat konvergálva eljut egy magas aktivációt kiváltó végső éredményhez. Ezt az eredményt gyakran megerősíti az ELBO, illetve a második legnagyobb aktiváció alacsnony értéke is. Oszcilláló, csökkenő és stagnáló aktivációk esetén az optimalizáció sikertelen. A 21. ábra példát mutat egy sikeres, míg a 23. ábra egy sikertelen optimalizáció kvantitatív elemzésére.

3.8.2. Kvalitatív validáció

A kvalitatív validációs technikák az optimalizált minta emberi értelmezhetőségét vizsgálják. Ezen esetekben, mivel nincs rá módszer, hogy meghatározzuk, hogy egy adott minta valóban a vizsgált egység által megtanult tulajdonságokat tükrözi, a gyakorlatban a különböző módszerek közötti konzisztenciára kell alapoznunk. A következő kvalitatív validációs technikákat alkalmaztam:

- annak vizsgálata, hogy a generált minta hasonlít-e a fehér-zaj analízis által előállítottra. Mivel a fehér-zaj analízis csak egy lineáris közelítést ad az egység receptív mezőjére, így az nem hordoz információt, ha az AM által generált mintánkkal ez nem mutat hasonlóságot. Amennyiben ezek között észrevehető valamilyen kapcsolat, akkor nagyobb bizonyossággal mondhatjuk, hogy egy megfelelő eredményre jutott az optimalizáció. Ez a kapcsolat a vizsgált egységről is azt sejteti, hogy annak az aktivációja valamilyen egyszerűbb lineáris kapcsolatban áll a bemenettel.

- annak vizsgálata, hogy a generált bemenet hasonlít-e a vizsgált, generatív modell képszintézisével előállított mintájára. Hasonlóan az előzőhöz, itt is az ezek közti kapcsolatok hordoznak információt.

- annak vizsgálata, hogy a generált minták mekkora varianciával rendelkeznek (1) egy adott optimalizációs konfiguráción belül, tehát mennyire befolyásolják a végeredményt a véletlen inicializáció és a mintavételezések; (2) egy adott AM módszeren belül mennyire érzékeny a módszerre jellemző hiperparaméterekre; illetve (3) módszereken átívelően, tehát milyen struktúrális különbségek vannak a vizsgált módszerek eredményei között.

3.9. MEI (Most Exciting Input) csomag

A kísérletek hatékony elvégzéséhez létrehoztam egy, a PyTorch keretrendszerre épülő csomagot. A csomag lehetőséget nyújt az összes ismertetett módszer és regularizációs technika alkalmazására. Az AM optimalizációt egy egyszerűen beállítható hiperparaméter konfiguráció alapján végzi. Fő tervezési szempont volt a rugalmasság, ennek eredményeként tetszőleges modell és művelet használható a célfüggvényben. Az optimalizációt követően számos analizációs lehetőséget nyújt a program. Ezek között van:

- az optimalizációs költségfüggény, az aktiváció és egyéb hasznos kvantitatív metrikák, illetve az aktuális kép iterációnkénti változásának a vizualizálása;

- lineáris közelítésen alapuló módszerek alkalmazása az optimalizálási célfüggvény maximalizálására (spike triggered analysis, optimális Gabor szűrő generálása, legjobb Gabor szűrő megtalálása egy rácskeresésben);

- egy adathalmazból aktivációban generált mintá(k)hoz legközelebb eső adatpontok megtalálása;

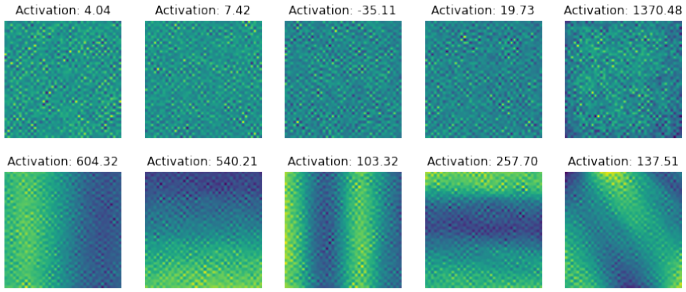
- a generált minta maszkolására, eltolására, jitter-elésére és az ezen transzformációk utáni aktivációk vizsgálatára

A kód elérhető ezen a linken: https://github.com/lacykaltgr/mei

4. Eredmények

4.1. Fehér zaj analízis

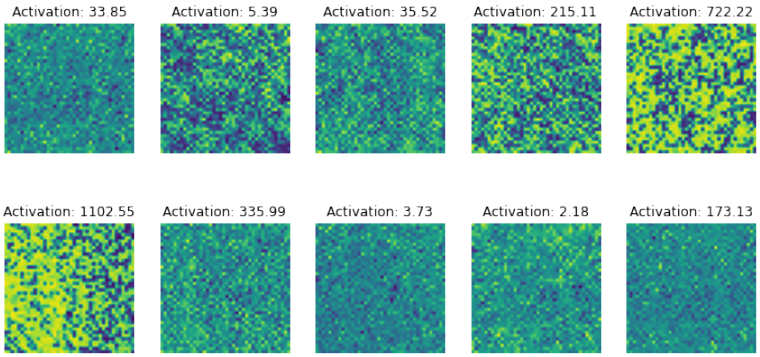
Az SPA analízissel szintetizált preferált bemenetek vizuálisan is egyértelműen megkülönböztetik a textúra-érzékeny és a nem textúra-érzékeny aktív dimenziókat (11. ábra). A nem-textúra-érzékeny dimenziók preferált bemenetein jól láthatóak az irányított élek, amelyek magas aktivációt váltanak ki a vizsgált egységből. A textúra-érzékeny dimenziók preferált bemenetei azonban ”semmitmondó” képeket hoznak létre, amelyeknek nincs lényegi magas szintű struktúrája, és amelyek aktivációja is alacsonyabb, mint a nem textúra-érzékeny dimenziók preferált bemenetein mértek. Ez igazodik az elvárásaimhoz, ugyanis a textúra-családok bírnak egyedi, nem-lineáris jellemzőkkel, így elmondható, hogy megkülönböztetésükhöz is az egység aktivációjának nem-lináris kapcsolatban kell állnia a bemenettel. Egy ilyen nem-lináris kapcsolat azonban nem vizualizálható az SPA módszerrel. A megközelítés egy másik hátránya, hogy a minták nagy mennyisége miatt az aktivációk mérése és átlagolása számításilag megterhelő és nem skálázódik jól nagyobb számítási erőforrásokat igénylő modellekre.



11. ábra. A fehér-zaj analízissel előállított vizualizációk és az ezekhez tartozó aktivációk a vizsgált egységeken. Felső sor: textúra-érzékeny dimenziók értelmezhetetlen zaj mintákat adnak; alsó sor: nem textúra-érzékeny dimenziók estében élminták láthatóak.

4.2. Pixeltér alapú AM

A pixel-térben való AM optimalizáció regularizáció nélkül zajos, konzisztens értelmezhető struktúrával nem rendelkező képeket eredményez (12. ábra). Az iterációszám növelésével ezek az eredmények egyre nagyobb aktivációt váltanak ki a vizsgált egységekből. Az optimalizáció tehát sikeres, de értelmezhetőségben nem javulnak az eredmények. A következőkben a regularizációs technikákat értékelem az elvégzésük helye szerint csoportosítva.



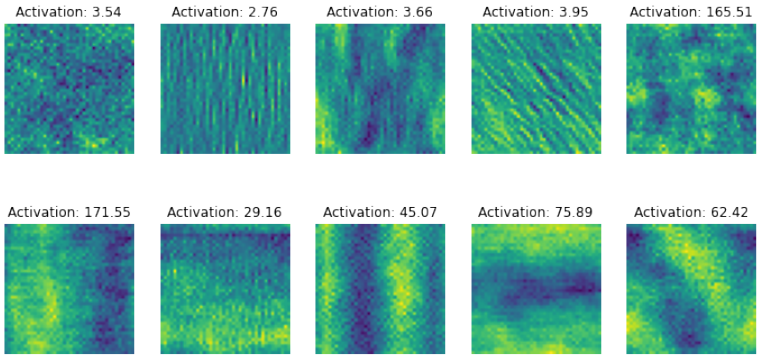
12. ábra. Pixel-térben való optimalizálás regularizáció nélkül magas-frekvenciás zaj képeket eredményez.

4.2.1. Optimalizációs lépés előtti regularizáció: $R_{\mathbf{pre}}$

– Véletlenszerű skálázás alkalmazásával a nem-textúra-érzékeny dimenziókra elkezdenek értelmezhető struktúrák megjelenni. Ezek bár emlékeztetnek a fehér-zaj analízissel kinyert preferált bemenetekre, még mindig sok magas-frekvenciás zaj jelenik meg a képen. Az eredmények nem fejlődtek a skálázás mértékének változtatásával és különböző ütemzésével sem.

– A bemenet az optimalizációs lépés előtti jitter-elése már mindkét csoportból magasabb aktivációkat kiváltó, jobban értelmezhető eredményeket ad (13. ábra). Ezek bár gyakran még mindig zajosak és elmosottak, de megjelennek összefüggő struktúrák a textúra-érézkeny dimenziók preferált bemenetein is. Kísérleteim alapján ez a regularizációs módszer kulcsfontosságú a pixeltér alapú optimalizációs beállításnál az értelmezhető eredmények eléréséhez.

– A bemenet normalizálása ebben a stádiumban nem mutatott lényegi változást az eredmények minőségében.

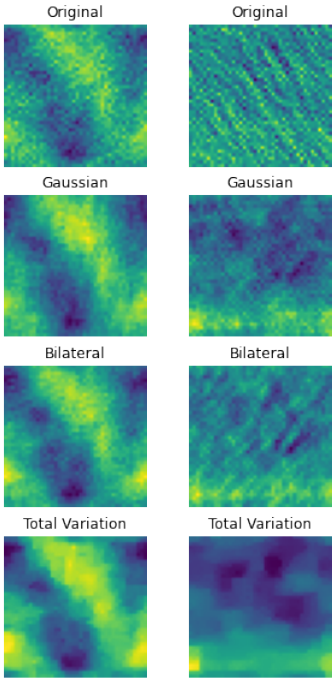


13. ábra. Az optimalizációs lépés előtti jitter-elés bevezetése lényegesen javít az eredményeken. Felső sor: textúra érzékeny dimenziók vizualizációin textúra-szerű mintázatok; alsó sor: a nem textúra-érzékeny egységeknél az eddig is látott élminták jelennek meg.

4.2.2. Optimalizációs lépés utáni regularizáció: $R_{\mathbf{post}}$

Az optimalizációs lépés utáni regularizációhoz zajtalanító műveletekkel kísérleteztem.

- **Gauss-elmosás** alapú iteráció végi elmosás redukálja a magas-frekvenciás zajt, de minőségileg nem javít lényegesen az eredményeken. A kompromisszum az elmosottság és a zaj között itt is megjelenik, ezt a szórással és a kernel mérettel tudjuk állítani. A textúra-érzékeny egységek esetében gyakran megnehezíti a tanulási folyamatot, így alacsony aktivációt, tehát sikertelen optimalizációt eredményezve.
- **Bilaterális szűrő** alkalmazásánál a Gauss-elmosáshoz hasonló jelenséget figyeltem meg. A textúra-érzékeny dimenzióknál ez is megakasztotta az optimalizációs folyamatot, így értelmezhetetlenné téve az szintetizált mintát.
- Egy másodlagos optimalizáció a **totális variancia** költségére szintén hatékonyan eltünteti a magas frekvenciás zajt. Az elmosottság azonban itt is zavaró mértékű, a szintetizált bemenetek "festmény-szerűek" lesznek. Az optimalizáció itt is gyakran sikertelen a textúra-érzékeny dimenziók esetében.

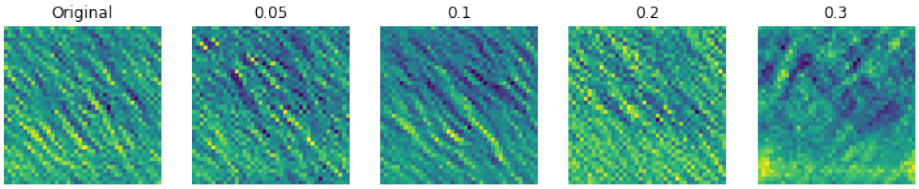


14. ábra. A különböző zajtalanítási eljárások jellemző hatásai a nem textúra-érzékeny (bal oszlop), illetve a textúra-érzékeny (jobb oszlop) egységek esetében.

Összességében a vizsgált optimalizációs lépés utáni regularizációs technikákról általánosságban elmondható, hogy nem javítanak lényegesen a vizualizációkon. Továbbá gyakran megnehezítik a konvergenciát, alacsony aktivációt kiváltó eredményekre vezetnek. Emellett újabb hiperparamétereket vezetnek be, amelyek megválasztása további kézi finomhangolást igényel.

4.2.3. Gradiens-kép regularizációja: $R_{\mathbf{grad}}$

A gradiensképben a magas-frekvenciás komponensek eltüntetéséhez egy Fourier-térben alkalmazott, alul-áteresztő, $1/\|f\|^{\gamma}$ átvitelű szűrőt használtam, ahol f jelöli a síkfrekvenciát, γ pedig a szűrési faktor, mellyel állítható a gradiens simításának a mértéke. Azt figyeltem meg, hogy ennek a szűrési faktornak túl magas értékre állítása értékes információt tüntet el a szintetizált bemenetből, azonban megfelelő értékű szűrőfaktorral koherensebb és stabilan kevésbé zajos, de nem túlságosan elmosott eredményekre jut az optimalizáció. Ez a jelenség a leginkább a textúra-érzékeny egységeknél figyelhető meg (15. ábra). Hozzá kell tenni viszont, hogy ez a hatás nem annyira látványos, mint a jitternél megfigyelt, így ez a reguralizációs módszer önmagában nem elég az értelmezhető eredményekhez.

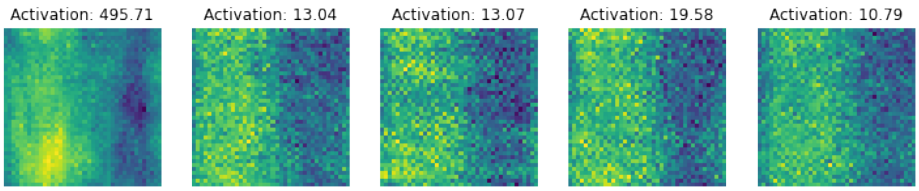


15. ábra. A gradienskép szűrésének jellemző hatása a textúra-érzékeny dimenziók vizualizációjára, különböző faktor (γ) értékek esetén. Túl kis γ esetén koherensebb, de zajosabb képeket eredményez, míg túl nagy értéke esetén eltűnnek a keresett textúra mintázatok (ebben az esetben az aktiváció is alacsonyabb).

4.2.4. Regularizáció a célfüggvényben: R_{obj}

Az optimalizációs célfüggényben az ELBO bevezetése ambivalenciát szül abban a tekintetben, hogy mégis mi az, amit keresünk. Egyik oldalról a tag olyan eloszlásba kényszeríti a bemeneteket, melyek pixel-intenzítás szempontjából nem rendelkeznek extrém értékekkel. Ez megnehezíti az optimalizációt, végeredményben kisebb aktivációkkal, több zajjal, és kevésbé kontrasztos struktúrákkal rendelkező szintetizált bemeneteket eredményezve (16. ábra). A magasabb értékű pixel-intenzításokat (és ebből következően magas aktivációkat) leszámítva struktúrában nem változtat lényegesen az eredményeken, azok mindössze zajosabbnak tűnnek. Elmondható viszont, hogy az ELBO bevezetése a költségbe csökkenti az optimalizáció iterációszámra és a lépésméretre való érzékenységét, ezzel stabilitást és konzisztenciát kölcsönözve az optimalizáció folyamatának.

A bemeneti kép **totális variancia** értékének hozzáadása a költséghez arra ösztönzi a képet, hogy azon egységes alakzatok rajzolódjanak ki. Emiatt hatékonyan tudja csökkenteni a bemeneteken megjelenő magas-frekvenciás zajt, azonban az egységes alakzatokra való kényszer túl hangsúlyos, alacsony aktivációkkal rendelkező képeket eredményez.



16. ábra. Az ELBO súlyozásának jellemző hatása a vizsgált egységekre és azok aktivációira (balról jobbra 0, 0.5, 1, 1.5, 2 értékű súlyozással). A súlyozás növelésével egyre kisebb a pixel-intenzítások varianciája, viszont egyre zajosabbak az eredmények.

4.2.5. Optimalizációs algoritmus

Megfigyeltem, hogy az optimalizációhoz használt algoritmus jelentős befolyással van az optimalizációs folyamat konvergenciájának sebességére és az eredményének minőségére (17. ábra).

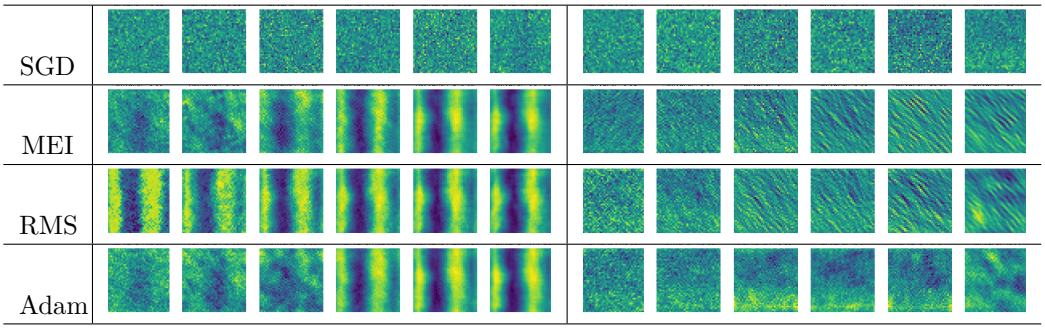
- a hagyományos **SGD** algoritmusban a bátorsági tényező nem igazodik semmilyen szinten a gradiens méretéhez. Emiatt az SGD optimalizáció érzékeny a bátorsági tényező és az iterációszám megválasztására, legtöbbször nem talál optimumot.

- a **MEI** algoritmusban a bátorsági tényező a gradiens abszolút értékének az átlagával fordítottan arányos, amely jelentősen javít a folyamat stabilitásán. A MEI algoritmussal már konzisztensen megjelennek koherens struktúrák. Ezek azonban még mindig elmosottak, de az iterációszám növelésével el lehet érni jobb minőségű mintákat.

- az **RMSprop** algoritmus már figyelembe veszi a korábbi gradiensek méretét is, így az előzőeknél lényegesen gyorsabb konvergenciát és jobb minőségű mintákat eredményez. Az iterációszám növelésével egyre nagyobb aktivációkat kiváltó mintákat talál.

- az **Adam** bővíti az RMSprop algoritmust a korábbi gradiensek exponenciális mozgóátlagát is figyelembe vevő momentum-mal. Ennek a bevezetése azonban megnehezíti az optimalizációt: a textúra-érzékeny dimenziókra nem talál optimumot, így azok vizsgálatára használhatatlan; a nem-textúra-érzékeny dimenziókra pedig lassabb a konvergenciája, mint az RMSprop-nak. Ennek az lehet az oka, hogy annyira erősen nem-konvex a felület, hogy az esetleges momentumos túllövések már más tartományba viszik az optimalizációt.

Összességében tehát az RMSprop kimagaslóan jobb eredményeket és gyorsabb konvergenciát ér el, mint a többi tesztelt algoritmus. A bátorsági tényező kézi ütemzése egyik esetben sem eredményezett érdemi javulást.



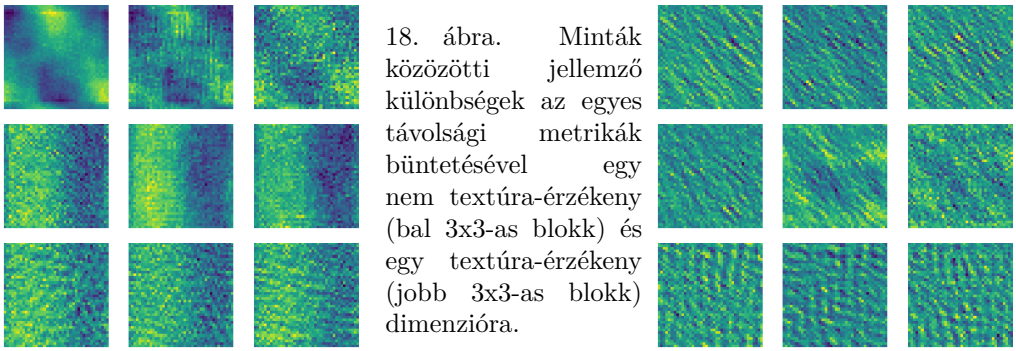
17. ábra. A vizualizált preferált bemenetek jellemző alakulása az iterációszám (oszlopok balról jobbra: 5, 10, 50, 100, 250) és az optimalizáció algoritmus választásának függvényében egy nem textúra-érzékeny (baloldali blokk), illetve egy textúra-érzékeny (jobboldali blokk) dimenzióra. Az RMSprop algoritmus lényegesen gyorsabban értelmezhető megoldást talál, mint alterantívái.

4.2.6. Változatosság

Megfigyelhető, hogy az AM optimalizáció a legtöbb esetben azonos, vagy nagyon hasonló végeredményre jut, ami elmondható mind globális, mint lokális szinten. Ez az egyik oldalról megnyugató, mivel meggyőződhetünk róla, hogy a vizualizált jellemzők egy általános optimumot mutatnak a pixeltérben való optimalizációs problémára, azonban nem tudjuk, hogy léteznek-e más optimumok. Hogy több belátást szerezzünk az egység működésébe bevezethetünk egy változatossági tényezőt a költségbe. Ez egy választott hasonlósági metrika alapján lehetőleg messze fogja kényszeríteni egymástól a szintetizált bemeneteket. Olyan minták kerestem, amik a lehető legnagyabb varianciát mutatják, de még rendelkeznek koherens struktúrával (18. ábra).

- a minták közötti **korreláció** büntetése nem eredményez lényegesen különböző képeket. Nem-textúra-érzékeny dimenzióknál ezek a különbségek gyakorlatban az eredeti kép zajosítását jelentik, tehát ezekre kifejezetten kártékony hatással van a korreláció büntetése. A textúra-érzékeny dimenzióknál az eddig is megfigyelt globális elrendezésbeli különbségeket erősíti fel a módszer. Az eredmények nagyon hasonlítanak az egységből kiváltott aktviációban is.
- a minták közötti **koszinusz hasonlóság** büntetésével sem tudunk eddig nem látott alakazatokat feltárni. A változatosság itt is főleg zaj, kontraszt és pixelintenzításbeli különbséget jelent, a minták struktúrájukban még mindig nagy hasonlóságot mutatnak. Aktivációkban viszont itt is kicsi a szórás egy-egy komponensre.
- az **euklideszi távolság** büntetésével a textúra-érzékeny komponensekre valamelyest eddigektől eltérő mintákat kapunk. Ez az eltérés azonban csak az eddigi vizsgálatok és az euklideszi távolságot büntető taggal való bővítés között mondható el, az optimalizáció ebben az esetben is hasonló bemeneti képeket ad. Elmondható, hogy az így előállított minták kisebb aktivációkat váltanak ki a vizsgált egységből és "hullám-szerű" (egy-egy síkfrekvencia által dominált) komponensek jelennek meg rajtuk.

Összességében elmondható ezekről a módszerekről, hogy egymással szorosan összefüggő elvi és gyakorlati problémákkal is rendelkeznek. Ezek: (1) csupán a változatosság kikényszerítésével nincsen konkrét irányításunk a szintetizált minták felett; (2) a változatossági tényező súlyozása újabb, nehezen hangolható hiperparamétert vezet be; és ettől függetlenül (3) a változatosságra kényszerített eredményekben nem jelennek meg lényegbeli különbségek, nem rendelkeznek lényeges új információval.



18. ábra. Minták közözötti jellemző különbségek az egyes távolsági metrikák büntetésével egy nem textúra-érzékeny (bal 3x3-as blokk) és egy textúra-érzékeny (jobb 3x3-as blokk) dimenzióra. Blokkokon belül fentről lefelé haladva a (1) korreláció; a (2) koszinuszos hasonlóság; és az (3) euklideszi távolság metrikához tartozó eredmények láthatóak.

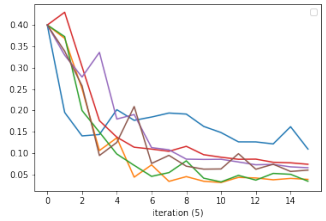
4.3. Eloszlás alapú AM

4.3.1. Tanulható szórással

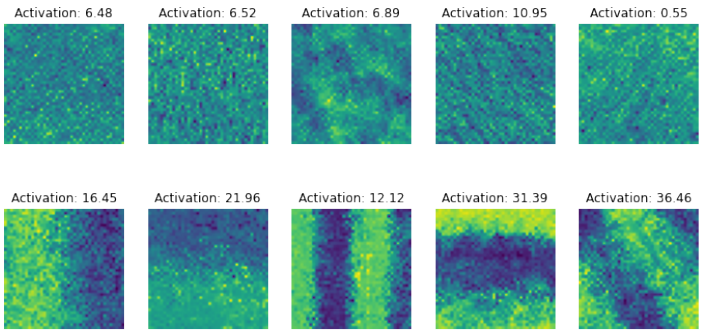
Egy egyszerű, két paraméteres (pl. normál, Laplace) eloszlás paraméterein végezve az optimalizációt azt feltételezem, hogy az adott egységhez tartozó preferált bemenetek ilyen eloszlás mintavételeinek tekinthetőek. Az eloszlást véletlen várható értékkel és szórással inicializálom, majd minden iterációban egy batch-et mintavételezek belőle. Az optimalizációs lépést minden iterációvan a batch akkumulált gradiensének az irányába végeztem.

- A nem-textúra-érzékeny dimenzióknál már minden regularizáció nélkül megjelennek a pixeltérben lévő optimalizációnál is eredményként kapott élképek, (20. ábra) azonban a minták még most is nagy mértékben zajosak. A bemenet optimalizációs lépés előtti jiterrel és a gradienskép szűrésével egyre élesebb és kontrasztosabb eredményekre jutottam, és az értelmezhető struktúrák eléréséhez szükséges iterációszám is csökkent. Ezek tehát a pixeltér alapú optimalizációnál is megfigyelt hatást hozzák.
- A textúra-érzékeny dimenzióknál erős regularizációt használva is elmondható, hogy bár a magas-frekvenciás komponensek emlékeztetnek a pixeltér alapú AM által szintetizálthoz, de hiányzik belőlük a koherencia. Az optimalizáció ennek ellenére sikeres, hiszen az eredmények magas aktivációkkal rendelkeznek, illetve az is elmondható, hogy pixeltérben végzettnél gyorsabban konvergálnak (21. ábra).

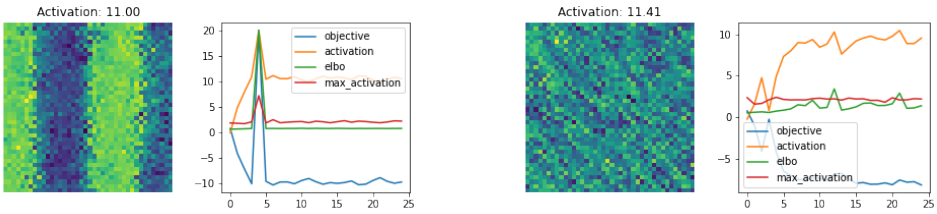
Megfigyelhető az is, hogy az eredmények minőségben és aktivációban is függetlenek az eloszlás választásától. Mind normál, mind Laplace eloszlás esetén hasonló várható értéket tanul meg az optimalizáció. Az iterációszám növelésével a tanulható szórások összeomlanak, tehát minden komponensre nulla közeli értéket vesznek fel. (19. ábra). Ez nem meglepő, hiszen az aktivációkat egy képpel a legegyszerűbb maximalizálni, nem egy azt terhelő valamilyen nem 0 szórású eloszlásból való mintavételekkel.



19. ábra. A szórások abszolút értékének átlaga nullához konvergál. Tehát a szórás nem tanul meg értékes információt.



20. ábra. A tanulható szórással végrehajtott eloszlás alapú AM vizualizációi. A textúra érzékeny dimenzióknál (felső sor) az alacsony a pixelek közötti korreláció ellenére kivehetőek a pixeltér alapú AM által generált textúrákhoz hasonló irányú komponensek. A nem textúra-érzékeny dimenziók (alsó sor) esetén az eddigi módszerek eredményeihez igazodó élminták jelennek meg.

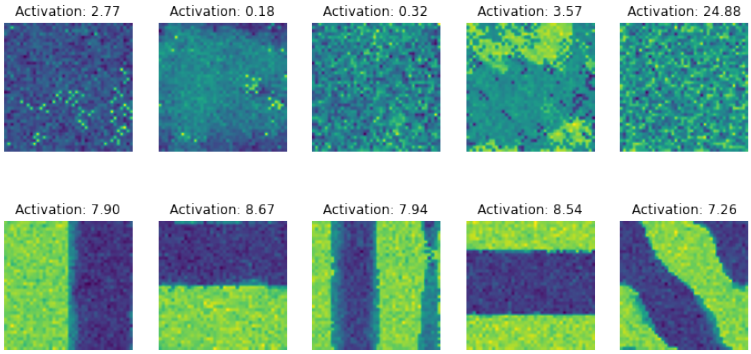


21. ábra. Validációs metrikák jellemző iterációnkénti vizualizációja, illetve a végeredmény egy nem textúra-érzékeny (bal oldal) és egy textúra-érzékeny (jobb oldal) egyésgre tanulható szórású eloszlás alapú AM esetén. Az optimalizáció mindkét esetben sikeres, hiszen: [1] nő az aktiváció (narancssárga); [2] az ELBO költsége (zöld) nulla közelében marad; [3] a vizsgált egységet kivéve a legnagyobb aktiváció mérete (piros) is alacsony marad; [4] az optimalizál célfüggvény (kék) értéke alacsony szintre konvergál.

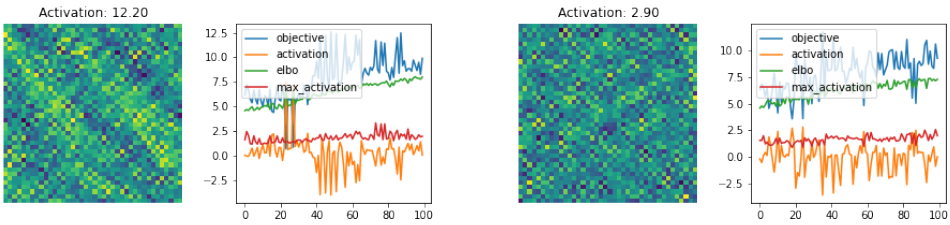
4.3.2. Rögzített szórással

Az eloszlás szórására egy rögzített érték beállítása esetén mintavételezéskor független és azonos eloszlású zajt adunk a várható értékéhez. Így az optimalizációval egy olyan értéket keresünk, amelyhez véletlen zajt adva is magas aktivációt tudunk kiváltani a vizsgált egységekből.

- A nem-textúra-érzékeny dimenziók preferált bemeneteire való optimalizáció már regularizációk nélkül is koherens és az előző megközelítésekhez képest kevésbé zajos eredményekre jut. A magas aktivációk és összefüggő struktúrák eléréséhez szükséges iterációszám még alacsonyabb, mint a tanulható szórás esetén. A bemenet jitter-elésével és a gradienskép egy magas szűrőfaktorú szűrésével a nem-textúra érzékeny dimenzióknál esztétikus, lényegében ”tökéletes”, minimális zajjal rendelkező eredményekre jutottam (22. ábra).
- A textúra-érzékeny dimenzióknál azonban a pixeltér alapú AM-hez képest lényegesen rosszabb minőségű mintákat talál a módszer. Ezeken csak ritkán és nagyon specifikus regularizációval jelennek meg koherens struktúrák, azok is gyakran alacsony aktivációkkal. Úgy tűnik, hogy itt is valamelyest érvényesül a fehér-zaj analízisnél megfigyelt hatás, miszerint zajok az aktivációval vett átlagához hasonlóan a zajok az aktivációra számolt gradiensenek koherenciát ösztönző komponensei ”kioltják” egymást (23. ábra).



22. ábra. Rögzített szórású, eloszlás alapú AM vizualizációi. A textúra-érzékeny dimenziók (felső sor) eredményeit alacsony aktivációk és rossz minőségű minták jellemzik. A nem textúra-érzékeny egységek (alsó sor) az eddig megfigyelteknél élesebb és kevesebb zajjal rendelkező eredményekre jutnak.

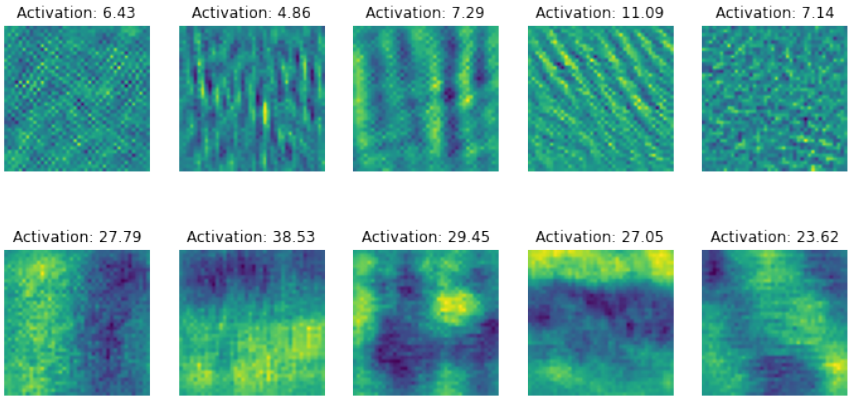


23. ábra. Rögzített szórású, eloszlás alapú AM vizualizációi és kvantitatív metrikáinak iterációnkénti változása két textúra-érzékeny egységre. A kiváltott aktiváció oszcillál ezért optimalizáció sikertelen. A baloldali egység tanult eloszlásának várható értéke azonban magas aktivációt vált ki a modellből, és struktúrában is emlékeztet a pixeltér alapú AM-nél látottakra.

A tanulható szórással való optimalizációhoz hasonlóan ebben az esetben is elmondható, hogy az eredmény invariáns az eloszlás választására. Bár a optimalizáció során egy eloszlást tanulunk, az ebből vett mintákat nézve a változatosság csak véletlenhez hasonló zaj hozzáadását jelenti a várható értékhez képest. Elmondható az is, hogy konfigurációkon belül és konfigurációk között is nagyon hasonló eredményekere jut a módszer. Így ebben az esetben sem beszélhetünk lényegi változatosságról a generált minták között. A célfüggvényben való regularizáció (ELBO, totális variancia) esetében azonos megfigyeléseket tettem, mint a pixeltér alapú AM esetében (4.2.4).
Az optimalizáló algoritmusok hatékonyságával kapcsolatban is hasonló megfigyeléseket tehetünk, mint amit a pixeltér alapú AM esetében is láthatunk. Az SGD továbbra sem ér el optimális eredményeket, míg a MEI bár kissé homályosabb eredményeket mutat, azok már általában koherens szerkezettel rendelkeznek, de alacsony aktivációkat eredményeznek. Az RMSprop lényegesen gyorsabb a többi algoritmusnál, és magas aktivációkat ér el. Az Adam algoritmus lassabban talál konvergenciára, mint a RMSprop, de valamelyest erősebben összefüggő struktúrával rendelkező és a vizsgált egységből magasabb aktivációt kiváltó mintákat generál. Ezt a változást a batch-tanítás következményének lehet tulajdonítani, ugyanis ezzel jobban megfontolt irányokba tesszük a gradiens lépéseket, így azok egyenletesebbek lesznek.

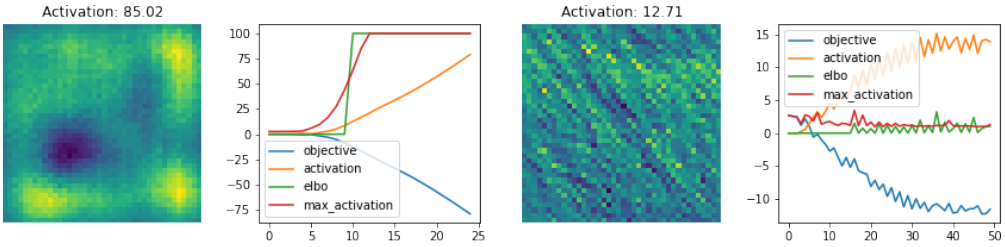
4.4. Transzformáció alapú AM

A transzformáció alapú AM esetén a bemeneten kívül egy tanulható paraméterekkel rendelkező műveletet is optimalizálok, ami megszűri a bemenetet a vizsgált modell előtt. Az ötlet, hogy az újonnan bevezetett paraméterek egy tanulható regularizációs folyamatot megvalósítva segítsék az optimum megtalálását. Gyakorlatban tanulható konvolúciós kerneleket használok. Első körben egy darab konvolúciós réteggel kísérletezem (ez 1 csatornás bemenet esetén mindössze 1 tanulható kernelt jelent), majd növelve ezek számát, az így kapott konvolúciós háló (CNN) regularizációs képességeit vizsgálom.
Elvárásaimhoz hűen az optimalizáció nagyon gyorsan konvergál egy magas aktivációt kiváltó bemenethez. Ezek a bemenetek azonban gyakran a vizsgált modellt "átverő" minták, és nem reprezentálják jól az egységek megtanult tulajdonságait. A pixel alapú AM-el ellentétben viszont ez nem inkoherens, zajos képeket, hanem a vizsgált modell tanult eloszlásán kívüli pixel-intenzítással rendelkező, de összefüggő, magas-szintű mintázatokat jelent. Az ELBO bevezetése a célfüggvénybe orvosolja ezt a helyzetet, azonban lassítja a konvergenciát, és gyakran bár magas aktivációt kiváltó, de még mindig nem egyértelműen értelmezhető mintákat szintetizál (25. ábra). Egyéb regularizációs műveletek alkalmazásával is próbálkoztam. Ezek közül a gradienskép szűrése jobban összefüggő és értelmezhető mintákhoz vezet (24. ábra).



24. ábra. A transzformáció alapú AM vizualizációi. A textúra-érzékeny dimenziók (felső sor) preferált bemenetei az eddig kinyerhetőknél nagyobb koherenciát mutatnak, a textúrák mintázatai már élesen kivehetőek. A nem textúra-érzékeny dimenziók esetén (alsó sor) a már ismert élmintákat láthatjuk.

Megfigyelhető az is, hogy a konvolúciós kernel méretének állításával szabályozhatjuk a megjelenő mintázatok kinézetét az aktviációkat magasan tartva. Ez a hatás a textúra-érzékeny dimenzióknál a mintázatok méretének és frekvenciájának a változtatását jelenti. A nem-textúra-érzékeny dimenzióknál egyre nagyobb kernel mérettel, egyre homályosabb lesz a szintetizált kép. Ez egy teljesen új megvilágításba helyezi a változatosság fogalmát, hiszen most először intuitíven irányíthatóvá válnak a generált minták. Ezt a hatást részletesebben a következő fejezetben fejtegetem, mivel még regularizáció alkalmazásával is elmondható, hogy a módszer bár képes a pixeltér alapú AM generált mintáitól különböző, jobb minőségű és magasabb aktivációkat kiváltó bemenetek szintetizálására, azonban nem garantálja ezeknek a megtalálását. Ehhez további regularizációra van szükség.



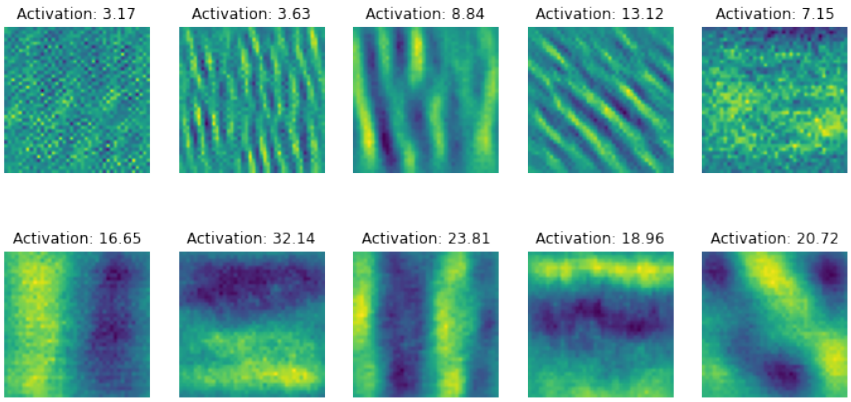
25. ábra. A transzformáció alapú AM egy jellemző iterációnkénti alakulása, illetve a végeredmény ELBO-val való regularizáció nélkül (jobb oldal) és ELBO-val való regularizációval (bal oldal) egy textúra-érzékeny dimenzióra.

Kettő vagy annál több konvolúciós réteggel az optimalizációs feladat tovább nehezedik, egyre jellemzőbb az "átverő" példák megtalálása. Itt már az ELBO-val való regularizáció sem eredményez az eddig megfigyeltekhez hasonló mintákat. Lassabb a konvergencia, és a vizualizációkon nagyon magas aktivációt kiváltó lokális foltok jelennek meg, de ezek sem stabilak még egy adott konfiguráción belül sem, így nem mondanak el értékes információt a modell működéséről.

4.5. Transzformáció + eloszlás alapú AM

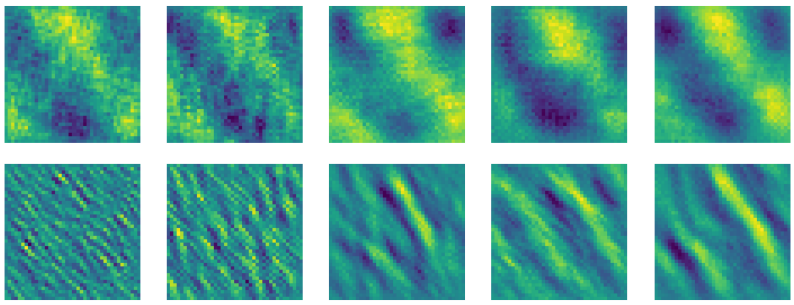
A konvolúciós műveletek új paramétereket vezettek be, amelyek megnehezítették egy értelmezhető optimum megtalálásának feladatát. A mérleg azonban egyensúlyba állítható újabb regularizáció bevezetésével. Új elvárás, hogy találjon az optimalizáció olyan paramétereket a transzformációs réteghez, amelyek robosztussá teszik a generált mintát a zaj ellen. Ehhez az eloszlás alapú AM által bevezetett ötletet alkalmazzuk: a transzformáció egy eloszlás paramétereit adja, majd ebből az eloszlásból mintavételezve kapjuk a vizsgált modell bemeneteit. Csak az egyrétegű konvolúció esetét részletezem, mivel a rétegek növelésével az előző fejezetben látott jelenség figyelhető meg.

- Tanulható szórás esetében az eloszlás alapú AM-hez hasonlóan (az egész optimalizáció intuitív értelmezésével konzistensen) a szórások összeomlanak, nem tanulnak meg értékes információt. Emiatt ettől a ponttól csak a nem tanulható szórás esetét tárgyalom.
- Rögzített szórással ez a módszer az eddig megfigyelteknél még gyorsabban egy értelmezhető és magas aktivációt kiváltó eredményre jut. Végre az is elmondható, hogy ezek konzisztensen optimumra jutnak egy konfiguráción belül (26. ábra). Ennek ellénére a pixel-térben való optimalizációval ellentétben ez nem feltétlen ugyanazt az optimumot jelenti. A gradienskép szűrésével még látványosabb mintázatok megtanulására késztethetjük az optimalizációt. Az ELBO-val való regularizáció ebben az esetben kifejezetten szükséges, teljesen elkerülhetőek alkalmazásával az "átverő" minták.



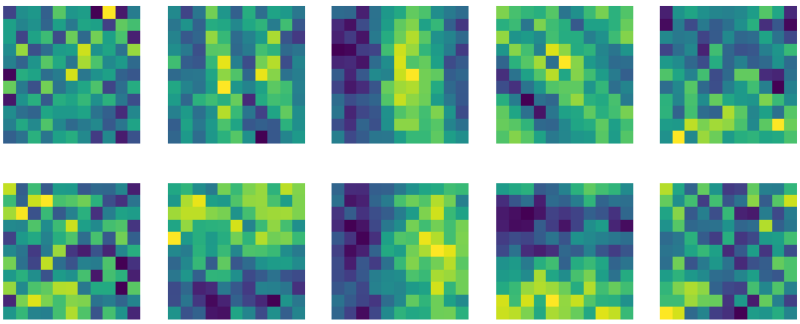
26. ábra. A transzformáció + eloszlás alapú AM-el kinyert vizualizációk. Az eredmények élesebb és koherensebb mintázatokat mutatnak, mint amit az eddig vizsgált egyéb megközelítések adtak mind a textúra-érzékeny (felső sor), mind a nem textúra-érzékeny (alsó sor) egységeknél.

A konvolúciós kernel méretének állításával itt is hasonló jelenség figyelhető meg, mint az egyszerű konvolúciós esetben. Irányítani tudjuk a minták jellemzőit, ami a módszer stabilitásával együtt egy, az eddigi megközelítésekben nem lehetséges "minta-család" szintetizálását teszi lehetővé egy adott egységre, így mélyebb belátást nyújtva annak működésébe (27. ábra).

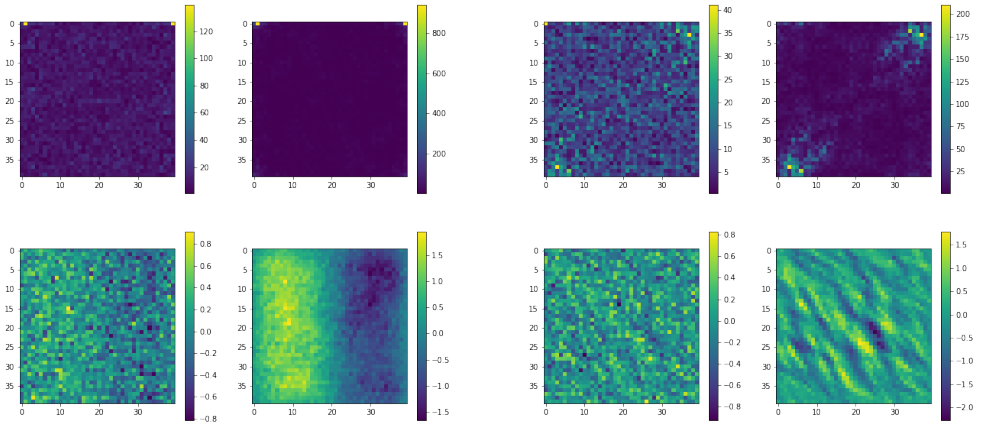


27. ábra. A tanulható konvolúciós kernel méretének jellemző hatása az eredményekre. Balról jobbra a kernelek méretei: 5, 7, 9, 11, 13. A nem textúra-érzékeny egységek esetén (felső sor) a kernel méret növelése kevésbé zajos, de esetekben jobban elmosott vizualizációkat eredményez. A textúra-érzékeny egységekhez tartozó (alsó sor) szintetizált bemenetek estében állítani tudjuk a megjelenő mintázatok méretét (a textúra alapharmonikusának a frekvenciáját).

Az optimalizált bemenet és konvolúciós kernel vizualizációjával (28. ábra) pedig a megtanult transzformációról is többet megtudhatunk. A bemeneten általában észrevehetőek az eddig látott vizualizációkhoz hasonló mintázatok, de ezek nem mutatnak koherenciát. Ugyanis az összefüggő alakzatokért a konvolúciós kernel felelős. Ezeket vizualizálva a textúra-érzékeny dimenziók esetében egy, az adott egységre jellemző textúra-részletet látható, míg a nem-textúra érzékeny dimenziók kernelei az eddig is megfigyelt élmintázatokra emlékeztetnek. Ennek az lehet az oka, hogy a bemenet egy fehér-zajhoz közeli jel, amely közel egyenletes teljesítménysűrűség spektrummal rendelkezik. Ha egy ilyen képet egy mintázattal szűrünk, akkor a képen megjelenő mintázatok amplitúdója megnő, így ezek lesznek láthatók a transzformált vizualizáción, míg elhelyezkedésüket a bemeneti jel fázis spektruma szabályozza. Tehát, mivel a kernel szabályozza a minta alakját, a transzformáció tényleg megfogja a magas szabadságfokú keresési problémát úgy, hogy olyan, alacsonyabb dimenziós altérbe kényszeríti azt, ami már valójában egy textúrához tartozó mintákat ír le. Úgy tűnik, hogy a kernel tanulja meg a mintázatokat, tehát a vizualizáció spektrumának amplitudóit, a bemeneti jel pedig a vizualizáció fázisát, tehát a mintázatok elhelyezkedését a képen. Ezt a feltételezést alátámasztja a bemenet, illetve a transzformált kép Fourier amplitudóspektrumainak a vizsgálata (29. ábra), hiszen látható, hogy az a bemeneti jelnél tipikusan egyenletesebb (magasabb entrópiájú), míg a szűrés kimenetén előálló képek esetén egyenetlenebb (kisebb entrópiájú). Tehát a bemeneti zajt a tanult transzformáció szűri meg interpretálható alakzattá.



28. ábra. A transzformáció + eloszlás alapú AM megtanult (11x11 méretű) konvolúciós kernelei. A textúra-érzékeny dimenziók (felső sor), és a nem textúra-érzékeny dimenziók (alsó sor) esetében is megjelnnek a végeredményen is látható magas szintű struktúrák.



29. ábra. Egy nem textúra-érzékeny (bal oldali blokk) és egy textúra-érzékeny (jobb oldali blokk) egység transzformáció + eloszlás alapú AM által megtanult bemenetének , transzformált bemenetének, illetve ezek Fourier amplitúdóspektrumainak vizualizációja (blokkon belül felső sorban a spektrumok, alsó sorban a képek, bal oszlopban a bemenetek, míg jobb oszlopban a transzformált bemenetek láthatóak). Látható, hogy a spektrum a bemeneti jelnél tipikusan egyenletesebb, míg a szűrés kimenetén előálló képek esetén egyenetlenebb. A bemeneti zajt a tanult transzformáció szűri meg interpretálható alakzattá.

A különböző optimalizálási algoritmusok máshogy viselkednek ebben az esetben, mint azt az eddigiekben láthattuk. A legstabilabb lefutásokat most a MEI algoritmus adja. Ez azt jelenti, hogy ebben az esetben az optimumok konzisztens megtalálásához szükség van a gradiens méretével való normalizálásra, azonban nem segítenek az előző iterációk gradienseinek értékei és normái. A többi algoritmussal ellentétben a MEI nem érzékeny a hiperparaméter beállításokra, az eddigieknél kevesebb kézi finomhangolást igényel a jó minőségű minták szintetizálása.

4.6. Tanult generátor modell alapú szintézis

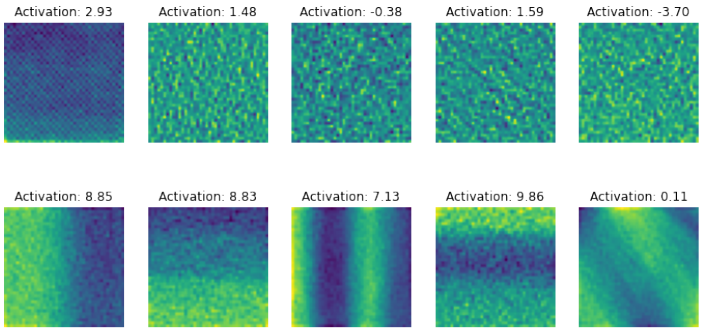
A generatív modellek további vizsgálatához kihasználhatjuk azt a tulajdonságukat, hogy képesek az általuk megtanult eloszlásból mintavételezni. Ezt több megközelítésből fogom alkalmazni a TDVAE vizsgált egységei által megtanult tulajdonságok vizualizálására. Az alábbi kísérleteket végzem el:

1. Skálázott one-hot vektorral való képszintézis. A modell generatív folyamatában $z2$ priorjából való mintavételezés helyett egy olyan kódból indulunk, ahol a vizsgált egység magasra, a többié pedig nulla értékűre van állítva.
2. AM optimalizáció $z2$ rejtett terében. Egy $k2$ kódot keresünk $z2$ terében, a mintákat pedig $z1$ és x priorjából való mintavételezésével állítjuk elő. Kísérletezem $z2$ priorjának inicializálására annak várható értékével, egy mintavételével, illetve skálázott one-hot vektor használatával. A one-hot vektoros inicializáció lényegében az előző pontban ismertetett folyamat eredményének további optimalizálását jelenti.
3. AM optimalizáció végzése $z1$ rejtett terében. Ebben az esetben egy $k1$ kódot keresek $z1$ terében, a mintákat pedig x priorjából való mintavételezéssel állítom elő. Az inicializáló kódok előállításához külön esetekben veszem $z2$ priorjának várható értékét, egy $z2$-ből való mintavételt, illetve egy skálázott one-hot vektort, majd az ezekből számított $z1$ prior várható értékét és mintavételét használom.

4.6.1. Képszintézis skálázott one-hot vektorból

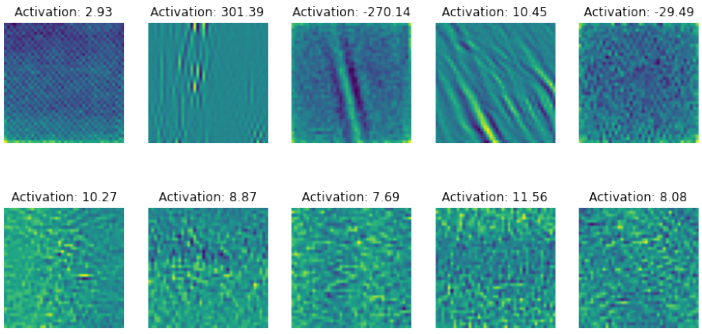
Az AM azt vizsgálja, hogy a generatív modellünk felismerő (recognition model, ENC) része bizonyos aktivációk kiváltásához milyen jellemzőket keres egy bemenetben. A skálázott one-hot vektorból való képszintézis ennek ellenében a generátor modell (DEC) viselkedését vizsgálja. A kérdés itt az, hogy magas aktivációkból milyen jellemzőket állít elő. A skálázás mértékével állíthatjuk, hogy milyen magas aktivációhoz szeretnénk mintákat szintetizálni. Skálázott one-hot vektorral kiszámítom $z1$ priorját, majd:

- Ennek a **várható értékét** veszem. A nem-textúra-érzékeny dimenzióknál már alacsony értékű skálázással is megjelennek az eddigi módszereknél is látható élmintázatok. Ezek egyre kontrasztosabbak és élesebbek a skálázás mértékének növelésével. A textúra-érzékeny dimenziók esetében azonban nem jelennek meg koherens struktúrák (30. ábra).



30. ábra. 10 értékűre skálázott one-hot vektorból való képszintézis eredményei a generatív modell priorjának várható értékét használva. A nem textúra-érzékeny egységek (alsó sor) aktivációi közel vannak a skálázási értékhez.

- Ennek ellenkezője történik $z1$ priorjából való **mintavételezéskor**. Ebben az esetben a textúra-érzékeny dimenziók mutatnak értelmezhető, az eddig megfigyeltekhez hasonló struktúrákat mutató eredményeket, és a nem-textúra-érzékeny egységeknél kaptam zajos mintákat (31. ábra).

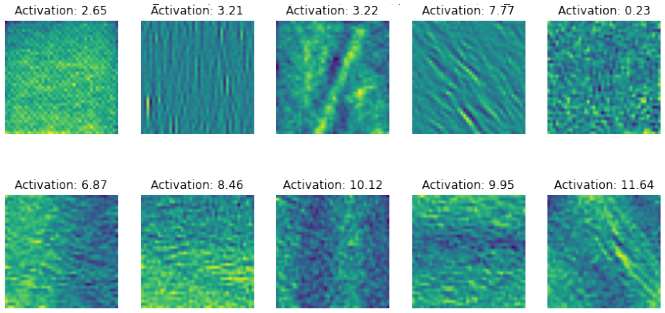


31. ábra. 10 értékűre skálázott one-hot vektorból való képszintézis eredményei a generatív modell priorjából mintavételezve.

Ez a megfigyelés elárulja, hogy $z1$-ben nem textúra-érzékeny egységek által eltárolt információt főleg a várható értékében, míg a textúra-érzékeny dimenziók által tárolt információt főleg a szórásban tárolja el.

4.6.2. Optimalizáció $z2$ terében

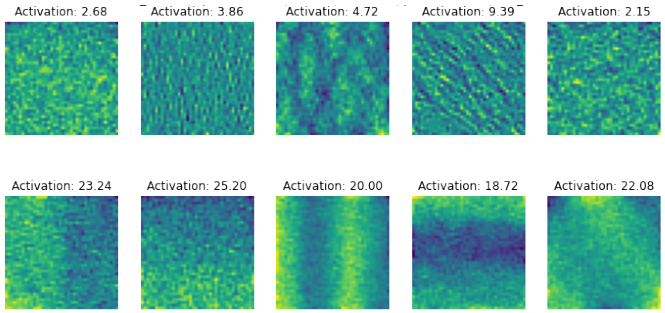
A nem textúra-érzékeny dimenziók vizualizációin sem a várható értékkel, sem egy mintavétellel inicializálva nem láthatóak a megszokott élminták, helyettük zajos, értelmezhetetlen képeket kapunk. A textúra-érzékeny dimenzióknál $z1$ priorjából mintát véve is esetenként jelennek csak meg koherens struktúrák. Bár az optimalizált kód általában várható módon megtanulja, hogy az éppen vizsgált egység dimenziójában lévő értéknek magasnak kell lennie, a vizualizációk mégis rossz minőségűek. A skálázott one-hot vektorral való inicializáció azonban felgyorsítja a konvergenciát, és jelentősen javít az eredményeken. A kód az egyszerű inicializálással is megtanulta, hogy magas értéket kell felvenni az adott dimenzióban. Ebben az esetben ennek az értéknek a már kezdésben magasra állításával hatékonyan segítjük egy optimum megtalálását. Az így kapott vizualizációk nagyon hasonlítanak az eddig vizsgált AM módszerek által generáltra (32. ábra).



32. ábra. A $z2$ térben való AM optimalizáció eredményei 5 értékűre skálázott one-hoz vektorből való inicializációval.

4.6.3. Optimalizáció $z1$ terében

A z1 terében való optimalizáció eredményét nem befolyásolja nagy mértékben az inicializáció választása. Mind a textúra-érzékeny, mind a nem-textúra-érzékeny dimenziók vizualizációja az eddig is tapasztalt eredményekre jut. Az iterációszám növelésével a generált minták egyre inkább hasonlítanak a pixeltér alapú AM eredményeihez. Ez nem meglepő, mivel $z1$ lineáris kapcsolatban áll a mintákkal. Különösen a nem textúra-érzékeny egységeknél az AM optimalizáció $z1$ terében stabilabban optimumot talál, mint a $z2$ terében. A textúra-érzékeny dimenziókra vizualizált mintázatok azonban kevésbé simák és koherensek (33. ábra).



33. ábra. A $z1$ térben való AM optimalizáció eredményei 5 értékűre skálázott one-hot vektorból inicializálva.

5. Összefoglalás

A munkám keretében módszereket javasoltam és dolgoztam ki olyan emberi interpretációra alkalmas minták generálására, amelyek bizonyos neuronok aktivációit maximalizálják. Ezeket az idegtudomány területén alkalmazva felhasználhatjuk az agy működésének mélyebb megértésére. Emellett rendkívül fontosak a neurális hálózatok értelmezhetőségének, robosztusságának és megbízhatóságának szempontjából is. Különösen érdekesek a generatív modellek ezen a téren, mivel lehetőséget kínálnak a generált bemeneteket a neurális hálózat által modellezni kívánt eloszlásba kényszeríteni.

Áttekintettem a szakirodalomban elérhető leginkább reprezentatív minták előállítására irányuló módszereket, és továbbfejlesztési javaslatokat tettem. Bemutattam, hogy a különböző optimalizációs algoritmusoknak és regularizációs technikáknak milyen hatása van a generált minták minőségére. A vizualizációk által bemutatott struktúrák és mintázatok egy helyes regularizációs konfigurációval jelentős hasonlóságot mutattak minden vizsgált módszer esetében.

Demonstráltam, hogy a tanulható regularizációs transzformáció nemcsak magasabb minőségű minták generálását teszi lehetővé, hanem lehetőséget nyújt az általa létrehozott alakzatok jellemzőinek hangolására is.
Javasoltam, hogy az optimalizálandó legjobban aktiváló bemeneteket egy eloszlással írjuk le. Ez a megközelítés lehetővé teszi több mintával egyszerre történő optimalizálást, ami gyorsabb konvergenciát eredményez, ugyanakkor bizonyos egységeknél nem értelmezhető eredményeket is produkálhat.
Az eloszlásos paraméterezés és a tanítható transzformáció ötvözése lehetővé teszi a többi vizsgált módszernél stabilabb és reprezentatívabb minták generálását.
A munkám egyik célja az volt, hogy megértsük a bemutatott módszerek működését is. Ehhez hipotéziseket fogalmaztam meg, amelyeket kísérletekkel minősítettem. Természetesen a dolgozatban csak az így alátámasztható hipotéziseket taglaltam.

A jövőbeli kutatásaim során a vizsgált modell látens változóinak mélyebb, általános értelmezésére összpontosítok. A top-down kontribúciók hatásait vizsgálom a tanult reprezentációkra, ami bepillantást nyújthat a látókéreg működésébe is. Ezenkívül érdekel, hogy az általam javasolt módszerek mennyire alkalmazhatóak általánosan, tetszőleges architektúrájú generatív neurális hálózatokban, illetve akár diszkriminatív célra tanított neurális hálózatokban.

Hivatkozások

[1] Andrea Asperti, Davide Evangelista, and Elena Loli Piccolomini. A survey on variational autoencoders from a green ai perspective. *SN Computer Science*, 2(4):301, 2021.

[2] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[3] Ali Borji and Sikun Lin. White noise analysis of neural networks. *arXiv preprint arXiv:1912.12106*, 2019.

[4] Ferenc Csikor, Balázs Meszéna, Bence Szabó, and Gergo Orban. Top-down effects in an early visual cortex inspired hierarchical variational autoencoder. In *SVRHM 2022 Workshop@ NeurIPS*, 2022.

[5] Rodrigo Echeveste, Laurence Aitchison, Guillaume Hennequin, and Máté Lengyel. Cortical-like dynamics in recurrent circuits optimized for sampling-based probabilistic inference. *Nature neuroscience*, 23(9):1138–1149, 2020.

[6] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[7] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. *Advances in neural information processing systems*, 28, 2015.

[8] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[9] Patrik Hoyer and Aapo Hyvärinen. Interpreting neural response variability as monte carlo sampling of the posterior. *Advances in neural information processing systems*, 15, 2002.

[10] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[11] Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven Siegelbaum, A James Hudspeth, Sarah Mack, et al. *Principles of neural science*, volume 4. McGraw-hill New York, 2000.

[12] Yan Karklin and Michael S Lewicki. Emergence of complex cell properties by learning to generalize in natural scenes. *Nature*, 457(7225):83–86, 2009.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[16] Tai Sing Lee and My Nguyen. Dynamics of subjective contour formation in the early visual cortex. *Proceedings of the National Academy of Sciences*, 98(4):1907–1911, 2001.

[17] Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent learning: Do different neural networks learn the same representations? *arXiv preprint arXiv:1511.07543*, 2015.

[18] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[19] Vasilis Marmarelis. *Analysis of physiological systems: The white-noise approach*.

Springer Science & Business Media, 2012.

[20] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. 2015.

[21] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4467–4477, 2017.

[22] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in neural information processing systems*, 29, 2016.

[23] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[24] Anh Nguyen, Jason Yosinski, and Jeff Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616*, 2016.

[25] Anh Nguyen, Jason Yosinski, and Jeff Clune. Understanding innovation engines: Automated creativity and improved stochastic optimization via deep learning. *Evolutionary computation*, 24(3):545–572, 2016.

[26] Anh Nguyen, Jason Yosinski, and Jeff Clune. Understanding neural networks via feature visualization: A survey. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 55–76, 2019.

[27] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

[28] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 3(3):e10, 2018.

[29] Bruno A Olshausen and David J Field. Natural image statistics and efficient coding. *Network: computation in neural systems*, 7(2):333, 1996.

[30] Gergő Orbán, Pietro Berkes, József Fiser, and Máté Lengyel. Neural variability and sampling-based probabilistic representations in the visual cortex. *Neuron*, 92(2):530–543, 2016.

[31] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and trends® in Optimization*, 1(3):127–239, 2014.

[32] Vivak Patel. On sgd's failure in practice: Characterizing and overcoming stalling. *arXiv preprint arXiv:1702.00317*, 2017.

[33] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

[34] Javier Portilla and Eero P Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International journal of computer vision*, 40:49–70, 2000.

[35] R Quian Quiroga, Leila Reddy, Gabriel Kreiman, Christof Koch, and Itzhak Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102–1107, 2005.

[36] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern*

Recognition (CVPR), pages 1–9, 2015.

[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[40] Mike Tyka. Class visualization with bilateral filters, 2016.

[41] Edgar Y Walker, Fabian H Sinz, Erick Cobos, Taliah Muhammad, Emmanouil Froudarakis, Paul G Fahey, Alexander S Ecker, Jacob Reimer, Xaq Pitkow, and Andreas S Tolias. Inception loops discover what excites neurons most using deep predictive models. *Nature neuroscience*, 22(12):2060–2065, 2019.

[42] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[43] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[44] A.M. Øygard. Visualizing GoogLeNet Classes. page 6, 2015.