**داکیومنت تکلیف شماره 6**

* **سوال 1**

**الف) دو مشکل اصلی که در شبکه‌های عصبی با تعداد زیادی لایه ممکن است رخ دهد عبارتند از:**

1. **ناپدید شدن گرادیان: اگر تعداد لایه‌ها بسیار زیاد باشد، مشتقات متوالی درهم ضرب شده و ممکن است با توجه به توابعی مانند تابع exponential (exp)، گرادیان‌ها به صفر نزدیک شوند. در نتیجه، وزن‌ها به صورت کم کم صفر میل می‌کنند و مدل دچار ناپایداری می‌شود.**
2. **انفجار گرادیان: اگر وزن‌ها بیش از حد بزرگ شوند، مشتقات متوالی در هم ضرب شده و عدد بسیار بزرگی به دست می‌آید که باعث ناپایداری مدل می‌شود. در این حالت، خطای (loss) بسیار بزرگی داشته و ممکن است به NaN (Not a Number) برسد.**

**این مشکلات مرتبط با انتقال گرادیان از لایه‌های عمیق شبکه به لایه‌های پایین‌تر یا بالعکس است و ممکن است منجر به عدم پیدا کردن بهینه‌های محلی یا سرعت آموزش کمتر شود.**

**مشکل ناپدید شدن گرادیان رخ می‌دهد وقتی که گرادیان در طول عملیات پس‌انتشار در شبکه عصبی به طور قابل توجهی کاهش می‌یابد. این مشکل معمولاً زمانی رخ می‌دهد که تابع فعال‌سازی در لایه‌های پنهانی از نوعی باشد که مقادیر بزرگی را به صفر یا به مقادیر بسیار کوچک نزدیک می‌کند، مانند تابع sigmoid یا تانژانت هایپربولیک.**

**در صورتی که وزن‌ها در لایه‌های پنهانی کمتر از یک باشند، با هر لایه جدید، مشتقات تابع فعال‌سازی در عملیات پس‌انتشار در ضرب شوند. این باعث می‌شود که مقادیر گرادیان به صورت توانی کاهش یابند و در نهایت به صفر نزدیک شوند. این مشکل موجب می‌شود که لایه‌های قبلی در شبکه تقریباً در تعیین وزن‌ها تاثیری نداشته باشند و آموزش به صورت ناکارآمدی ادامه یابد. به عبارت دیگر، اطلاعات ضروری برای آموزش به لایه‌های پایین‌تر به درستی منتقل نمی‌شود و نتیجه‌ای کسب نمی‌شود.**

**مشکل انفجار گرادیان در واقع عکس مشکل ناپدید شدن گرادیان است. در این حالت، گرادیان به طور قابل توجهی افزایش می‌یابد و می‌تواند به مقادیر بسیار بزرگ برسد. این مشکل معمولاً زمانی رخ می‌دهد که وزن‌های شبکه بسیار بزرگ شوند یا توابع فعال‌سازی غیرخطی مانند ReLU (Rectified Linear Unit) استفاده شود.**

**در این حالت، هنگام محاسبه گرادیان در عملیات پس‌انتشار، ضرب متوالی در وزن‌های بزرگ ممکن است باعث افزایش نمودن گرادیان شود و به مقادیر بسیار بزرگی برسد. این موضوع می‌تواند منجر به نویزی شدن شبکه شود و آموزش را به همراه تغییرات ناپایدار در وزن‌ها از دست بدهد.**

**برای حل مشکل ناپدید شدن گرادیان، روش‌هایی مانند استفاده از توابع فعال‌سازی جدید مانند ReLU استفاده از روش‌های نرمال‌سازی وزن‌ها (مانند نرمال‌سازی چرخشی وزن‌ها) مورد استفاده قرار می‌گیرند.**

**به علاوه، برای مقابله با مشکل انفجار گرادیان، می‌توان از روش‌هایی مانند کاهش نرخ یادگیری (learning rate) و استفاده از روش‌های کاهش گرادیان مانند کاهش گرادیان جزئی یا Partial Gradient استفاده کرد.**

**ب) قبل از معرفی شبکه ResNet، شبکه‌های عمیق‌تر دچار مشکلاتی می‌شدند که به ناپدید شدن گرادیان و یا انفجار گرادیان منجر می‌شد. این مشکلات به دلیل وجود تعداد زیادی لایه در شبکه‌ها و اعمال توابع غیرخطی مانند تابع فعال‌سازی بود. این مشکلات باعث می‌شدند که در آموزش شبکه‌های عمیق، گرادیان به صفر نزدیک شده یا به مقادیر بسیار بزرگی افزایش یابند که باعث ناپایداری در آموزش مدل می‌شد.**

**شبکه ResNet این مشکلات را با استفاده از بلوک‌های residual به طور کامل برطرف کرد. بلوک residual به صورت متوالی تکرار می‌شود و هر بلوک شامل چندین لایه است. این بلوک‌ها عملکردی مشابه با تابع identity (تابع هویزن) دارند.**

**در هر بلوک، ورودی بلوک با خروجی بلوک قبلی جمع می‌شود و سپس به تابع فعال‌سازی انتقال داده می‌شود. این عملکرد به صورت shortcut یا connection skip شناخته می‌شود. از طریق این اتصال مستقیم، مقدار identity (ضرایب لایه‌های قبلی) حفظ می‌شود و به صورت تابع توانی با پایه کمتر از یک پیشرفت نمی‌کند. به این ترتیب، اگر ضرایب و بایاس‌ها در لایه‌های میانی به صفر برسند، با اضافه کردن مقدار قبلی به لایه‌های بعدی، از ناپدید شدن گرادیان جلوگیری می‌شود.**

**به عبارت دیگر، اگر در الیه‌های میانی، ضریب‌ها و بایاس‌ها به صفر برسند، با انتقال مستقیم مقدار قبلی به لایه‌های بعدی قبل از تابع فعال‌سازی، مانع از تولید خروجی صفر می‌شود و در نتیجه گرادیان که صفر شده است، تاثیری در به‌روزرسانی ضرایب خروجی نخواهد داشت. با این روش، مشکل ناپدید شدن گرادیان که در شبکه‌های عمیق قبلی وجود داشت، حل شده و امکان آموزش شبکه‌های با تعداد بسیار زیادی لایه وجود دارد.**

* **سوال 2**

**الف) ماژول مورد نظر شامل تصویر سه کاناله با ابعاد n × n است. برای محاسبه تعداد پارامترهای قابل آموزش و میدان تأثیر (field receptive) مربوط به این ماژول، به صورت زیر عمل می‌کنیم:**

**میدان تأثیر (field receptive): میدان تأثیر یعنی هر پیکسل خروجی به چند پیکسل ورودی وابسته است و ماژول ما نشان‌دهنده‌ی این میدان تأثیر است. بنابراین، به ترتیب شاخه‌های مختلف ماژول را بررسی می‌کنیم:**

1. **شاخه سمت چپ: این شاخه فقط از یک فیلتر 1 × 1 استفاده می‌کند. در نتیجه، هر پیکسل خروجی فقط از پیکسل متناظر خود در ورودی تأثیر می‌پذیرد و فیلد آن برابر با 1 × 1 است.**
2. **شاخه وسط: این شاخه دارای یک فیلتر 1 × 1 و یک فیلتر 3 × 3 است. فیلتر 1 × 1 هیچ تأثیری بر روی میدان تأثیر ندارد. اما فیلتر 3 × 3 باعث می‌شود میدان تأثیر این شاخه به ابعاد 3 × 3 برسد.**
3. **شاخه سمت راست: این شاخه دو فیلتر 3 × 3 دارد. طبق الگوی مشاهده شده در معماری VGG، استفاده از دو کانولوشن 3 × 3 باعث می‌شود میدان تأثیر این شاخه به ابعاد 5 × 5 برسد.**

**با توجه به این تحلیل، در نهایت میدان تأثیر کلی ماژول برابر با 5 × 5 خواهد بود.**

**بلوک موردنظر شامل چندین لایه کانولوشنی است که از تصویر ورودی سه کاناله n × n استفاده می‌کند. برای محاسبه تعداد پارامترها، از فرمول ((m \* n \* d) + 1) \* k) استفاده می‌کنیم. در این بلوک، لایه‌های کانولوشنی با ابعاد 1 در 1 و 3 در 3 وجود دارند. همچنین، یک لایه کانولوشنی 1 در 1 و 3 در 3 با عمق 32 و یک لایه کانولوشنی 1 در 1 با عمق 96 نیز وجود دارد.**

**تعداد پارامترها برای هر لایه به شرح زیر است:**

* **لایه کانولوشن 1 در 1 با عمق 32: 128 پارامتر**
* **لایه کانولوشن 3 در 3 با عمق 32: 9248 پارامتر**
* **لایه کانولوشن 3 در 3 با عمق 32: 9248 پارامتر**
* **لایه کانولوشن 1 در 1 با عمق 96: 128 پارامتر**
* **لایه کانولوشن 3 در 3 با عمق 32: 9248 پارامتر**
* **لایه کانولوشن 1 در 1 با عمق 96: 128 پارامتر**
* **لایه کانولوشن 1 در 1 با عمق 256: 24,832 پارامتر**

**پس از جمع تمامی این مقادیر، تعداد کل پارامترهای قابل آموزش در بلوک موردنظر برابر با 52960 پارامتر است.**

**ب)**

**حالت A:**

**Conv2D(filters=16, kernel\_size=(3, 3), padding='valid')**

**Conv2D(filters=32, kernel\_size=(3, 3), padding='valid')**

**در لایه‌های Conv2D، تعداد پارامترهای قابل آموزش به صورت زیر محاسبه می‌شود:**

**تعداد پارامترها = (تعداد کانال‌های ورودی) × (تعداد فیلترها) × (سایز کرنل) + (تعداد فیلترها)**

**برای لایه اول Conv2D:**

**تعداد پارامترها = 3 × 16 × (3 × 3) + 16 = 448**

**برای لایه دوم Conv2D:**

**تعداد پارامترها = 16 × 32 × (3 × 3) + 32 = 4640**

**بنابراین، مجموع تعداد پارامترهای قابل آموزش در حالت A برابر است با 448 + 4640 = 5088.**

**برای محاسبه زمینه اثر (receptive field)، باید سایز کرنل و پدینگ (padding) هر لایه Conv2D را در نظر بگیریم. زمینه اثر یک لایه، منطقه‌ای از تصویر ورودی است که بر روی یک پیکسل خروجی تأثیر می‌گذارد.**

**برای لایه اول Conv2D با سایز کرنل (3، 3) و پدینگ 'valid'، زمینه اثر 3 در 3 است و برای لایه دوم Conv2D با سایز کرنل (3، 3) و پدینگ 'valid'، زمینه اثر نیز 3 در 3 است. بنابراین، زمینه اثر حالت A برابر است با 3 در 3.**

**حالت B:**

**LocallyConnected2D(filters=16, kernel\_size=(3, 3), padding='valid')**

**LocallyConnected2D(filters=32, kernel\_size=(3, 3), padding='valid')**

**در لایه‌های LocallyConnected2D، تعداد پارامترهای قابل آموزش به صورت مشابه لایه‌های Conv2D محاسبه می‌شود. با این حال، لایه‌های LocallyConnected2D برای هر پیکسل ورودی وزن متفاوتی دارند که منجر به تعداد بیشتری از پارامترها می‌شود.**

**برای لایه اول LocallyConnected2D:**

**تعداد پارامترها = (تعداد کانال‌های ورودی) × (تعداد فیلترها) × (اندازه هسته) × (اندازه هسته) + (تعداد فیلترها)**

**برای لایه دوم LocallyConnected2D:**

**تعداد پارامترها = 16 × 32 × (3 × 3 × 3) + 32 = 13856**

**بنابراین، مجموع تعداد پارامترهای قابل آموزش در حالت B برابر است با 13856.**

**زمینه تاثیر لایه‌های LocallyConnected2D معادل سایز کرنل است، به همان شکل لایه‌های Conv2D. بنابراین، زمینه تاثیر حالت B نیز 3 در 3 است.**

**در نتیجه:**

**- حالت A شامل 5088 پارامتر قابل آموزش و زمینه تاثیر 3 در 3 است.**

**- حالت B شامل 13856 پارامتر قابل آموزش و زمینه تاثیر 3 در 3 است.**

**برای تعیین زمینه تأثیر در هر یک از حالت‌ها، باید سایز کرنل و پدینگ لایه‌های کانولوشنال را در نظر بگیریم.**

**در هر دو حالت A و B، اندازه هسته (3، 3) است که نشان می‌دهد یک فیلتر 3 در 3 استفاده می‌شود. این بدان معناست که هر پیکسل خروجی توسط یک همسایگی 3 در 3 پیکسلی از تصویر ورودی تأثیر می‌پذیرد.**

**علاوه بر این، هر دو حالت از پدینگ 'valid' استفاده می‌کنند که بدان معناست که هیچ پدینگی به تصویر ورودی اضافه نمی‌شود. با استفاده از پدینگ 'valid'، لایه‌های کانولوشنال تنها پیکسل‌های درون مرزهای تصویر ورودی را در نظر می‌گیرند و زمینه تأثیر آن‌ها نسبت به پدینگ 'same' یا 'causal' کوچکتر خواهد بود.**

**بنابراین، در هر دو حالت A و B، زمینه تأثیر 3 در 3 است. این بدان معناست که هر پیکسل خروجی توسط یک محیط محلی 3 در 3 در تصویر ورودی تحت تأثیر قرار می‌گیرد.**

**لازم به ذکر است که زمینه تأثیر اندازه ناحیه محلی است که هر لایه در آن هنگام اعمال عملیات خود را در نظر می‌گیرد. زمینه تأثیر بزرگتر امکان درک وابستگی‌های سراسری بیشتری را به مدل می‌دهد، در حالی که زمینه تأثیر کوچکتر بیشتر به جزئیات محلی تمرکز می‌کند.**

* **سوال 4**

**الف) مفهوم stride یا گام در لایه های کانولوشنی مربوط به تعداد پیکسل‌هایی است که فیلتر در هر مرحله به طول و عرض ورودی حرکت می‌کند. با تعیین مقدار stride می‌توان فاصله بین هر مرحله از کانولوشن را تعیین کرد. در واقع، stride نشان می‌دهد چقدر فاصله بین محاسبات کانولوشنی است.**

**تفاوتی که stride با Pooling دارد این است که در stride اطلاعات میانگین یا حداکثر بین هر مرحله از کانولوشن منتقل نمی‌شوند و فقط محاسبات کانولوشنی انجام می‌شود. از طرف دیگر، در Pooling، مقادیر میانگین یا حداکثر در هر مرحله از کانولوشن گرفته شده و اطلاعات فشرده‌تری از تصویر بدست می‌آید.**

**تأثیر stride در عملکرد شبکه‌های عصبی این است که با افزایش مقدار stride، ابعاد خروجی کانولوشن کاهش می‌یابد و تعداد پارامترها نیز کاهش می‌یابد. این می‌تواند کمک کند تا مدل سریع‌تر آموزش ببیند و پیچیدگی محاسباتی را کاهش دهد. با این حال، با افزایش مقدار stride، اطلاعات موجود در تصویر کاهش می‌یابد و ممکن است بخشی از اطلاعات مهم را از دست بدهیم. بنابراین، باید توازنی بین کاهش پارامترها و حفظ اطلاعات مهم برقرار کرد تا عملکرد شبکه بهینه شود.**

**ب)**

**ب1) برای لایه های میانی می‌توان از تابع فعال‌سازی ReLU (Rectified Linear Unit) استفاده کرد. تابع ReLU از آنجا که فقط مقادیر مثبت را انتقال می‌دهد و مقادیر منفی را به صفر تبدیل می‌کند، به خوبی در شبکه‌های عصبی عمل می‌کند. این تابع محاسباتی ساده و سریع دارد و باعث افزایش تنش و عملکرد غیرخطی مدل می‌شود.**

**ب2) با توجه به این که مسئله به صورت باینری است، یعنی باید یک برچسب دوتایی را پیش‌بینی کنیم، تابع خطای مناسب برای این مسئله می‌تواند تابع باینری کراس-انتروپی (Binary Cross-Entropy) باشد. دلیل انتخاب این تابع خطا عبارت است از:**

1. **تابع باینری کراس-انتروپی یک تابع خطا مناسب برای مسائل دسته‌بندی باینری است. با استفاده از این تابع، می‌توانیم اختلاف بین توزیع احتمالی خروجی شبکه و توزیع احتمالی مورد انتظار (برچسب واقعی) را اندازه‌گیری کنیم. این تابع خطا مقدار کمتری به عنوان خروجی می‌دهد اگر توزیع احتمالی خروجی شبکه به درستی با توزیع احتمالی مورد انتظار همخوانی داشته باشد و مقدار بیشتری در صورت عدم تطابق بین این دو توزیع.**
2. **تابع باینری کراس-انتروپی معمولاً با تابع فعال‌سازی سیگموید (Sigmoid) در لایه خروجی شبکه استفاده می‌شود. تابع سیگموید مقادیر خروجی را بین 0 و 1 محدود می‌کند که به خوبی برای مسئله باینری مناسب است.**
3. **این تابع خطا دارای خواص محاسباتی مناسبی است و به خوبی قابل بهینه‌سازی است. در روش‌های بهینه‌سازی مانند نزول تصادفی گرادیانی (Stochastic Gradient Descent) و الگوریتم‌های پیشرفته‌تر، می‌توان از تابع باینری کراس-انتروپی به عنوان تابع هدف استفاده کرد و بهینه‌سازی شبکه را با توجه به آن انجام داد.**

**بنابراین، با توجه به ماهیت مسئله باینری و استفاده از تابع فعال‌سازی سیگموید، تابع باینری کراس-انتروپی به عنوان تابع خطا مناسبی برای آموزش شبکه در این مسئله به نظر می‌رسد.**

**ب3) در این مسئله، اگر بخواهیم دقت مدل در تشخیص محصولات معیوب از محصولات سالم زیاد باشد و همچنین تعداد محصولات معیوبی که به دست مشتری می‌رسد را به حداقل ممکن کاهش دهیم، می‌توانیم عملکرد مدل را با استفاده از معیار Precision بیان کنیم.**

**دلیل انتخاب Precision عبارت است از:**

1. **Precision نسبت True Positive (TP) به جمع True Positive و False Positive (FP) را نشان می‌دهد. به این معنی است که Precision نسبت معیوب‌های شناسایی شده درست به کل محصولاتی است که مدل به عنوان معیوب شناسایی کرده است. با افزایش Precision، تعداد محصولات سالمی که اشتباهاً به عنوان معیوب شناسایی می‌شوند (False Positive) کاهش می‌یابد.**
2. **در این مسئله، ممکن است تشخیص محصولات معیوب با دقت بالا مهم‌تر از این باشد که تمام محصولات معیوب را تشخیص دهیم. به عبارت دیگر، می‌توانیم برای جلوگیری از رسیدن محصولات معیوب به دست مشتری، تعداد محصولات سالمی که به اشتباه به عنوان معیوب شناسایی می‌شوند را قبول کنیم. در این صورت، با افزایش Precision، احتمال رسیدن محصولات معیوب به دست مشتری (False Negative) کاهش می‌یابد.**

**بنابراین، با در نظر گرفتن دقت مدل و کاهش تعداد محصولات معیوبی که به دست مشتری می‌رسد، معیار Precision مناسب‌تر از Recall به نظر می‌رسد.**

**ج)  
ج1) شبکه‌های عصبی کانولوشنی (CNN) برای طبقه‌بندی موضوع متن موفق نخواهند بود. دلیل این امر به دو عامل برمی‌گردد. اولاً، CNN ها برای ورودی‌هایی با سایز محدود طراحی شده‌اند، در حالی که متن‌ها می‌توانند اندازه‌های متفاوتی داشته باشند. این باعث می‌شود که ساختار شبکه کانولوشنی برای پردازش کلمات و جملات متفاوت مشکلاتی داشته باشد. ثانیاً، در متن، معنای یک کلمه نسبت به جمله کلی به موقعیت مکانی واژه‌ها بستگی دارد، در حالی که شبکه‌های کانولوشنی برای تشخیص معناهای مکانی مناسب نیستند. بنابراین، در طبقه‌بندی موضوع متن، شبکه‌های کانولوشنی مناسب نیستند.**

**ج2) در تشخیص گوینده از روی صوت، شبکه‌های عصبی کانولوشنی (CNN) می‌توانند موفق باشند. با تبدیل داده ورودی (صدا) به شکل‌های خاصی مانند مجموعه ویژگی‌های تبدیل فرکانس-فرکانس سریع (MFCCs)، می‌توان اطلاعات صوتی را به صورت یک بردار دو بعدی مورد بررسی قرار داد. این ویژگی‌ها به مدل CNN اجازه می‌دهند تا الگوهای مرتبط با گوینده را تشخیص داده و تشخیص دهنده صوتی را با دقت بالا ارائه دهند.**

**ج3) به علت عدم وجود رابطه فضایی بین ورودی‌ها، احتمالاً شبکه‌های عصبی کانولوشنی (CNN) در این کاربرد به خوبی عمل نمی‌کنند. در اینجا می‌توان از روش‌های مبتنی بر سری زمانی مانند شبکه‌های LSTM استفاده کرد که به طور دقیق این موضوع را بررسی می‌کنند. به طور کلی، چون در این مورد هیچ ارث‌بری وجود ندارد، احتمالاً نمی‌توان از شبکه‌های عصبی کانولوشنی استفاده کرد. به عبارت دیگر، عملکرد آن‌ها به صورت کلی الگوی خاصی را به وجود نمی‌آورد.**

**د) شبکه‌های عصبی کانولوشنی می‌توانند با مشکلات و چالش‌های زیر روبرو شوند:**

1. **نیاز به حجم بالای داده: برای آموزش یک شبکه عصبی کانولوشنی قوی و دقیق، نیاز به حجم بالایی از داده‌های آموزشی است. در صورت عدم دسترسی به داده‌های کافی، ممکن است شبکه به‌درستی عمل نکند و دقت آن کاهش یابد.**
2. **پارامترهای زیاد: شبکه‌های کانولوشنی بسیار پارامترهای قابل تنظیمی دارند که نیاز به تنظیم و آموزش مناسب دارند. مدیریت و بهینه‌سازی این پارامترها ممکن است زمان‌بر و پیچیده باشد.**
3. **مشکل overfitting: در صورتی که شبکه کانولوشنی بیش‌برازش (overfitting) داشته باشد، دقت آن بر روی داده‌های آموزشی بسیار بالا خواهد بود اما بر روی داده‌های جدید ناشناخته ممکن است نتایج نامطلوبی را نشان دهد. کنترل و مدیریت overfitting نیازمند استفاده از روش‌های مناسبی مانند dropout و regularization است.**
4. **نیاز به قدرت پردازشی بالا: آموزش و استفاده از شبکه‌های کانولوشنی به دلیل پردازش ماتریس‌های بزرگ و عملیات مکرر ممکن است نیاز به قدرت پردازشی بالا و استفاده از سخت‌افزارهای پر قدرت را داشته باشد.**
5. **حساسیت به تغییرات ورودی: شبکه‌های کانولوشنی در مواجهه با تغییرات کوچک در ورودی ممکن است عملکرد ضعیفی داشته باشند. به عنوان مثال، اگر یک تصویر را به طور کوچکی جابه‌جا کنیم، خروجی شبکه ممکن است به طور قابل توجهی تغییر کند.**
6. **محدودیت در فهم معنای دقیق: شبکه‌های کانولوشنی بیشتر برای تشخیص الگوها و ویژگی‌های مکانی در داده‌ها استفاده می‌شوند. بنابراین، در فهم معنای دقیق و مفاهیمی که بر مبنای زبان و دستور زبانی هستند، ممکن است نقص داشته باشند.**
7. **انتقال یادگیری محدود: در برخی موارد، شبکه‌های کانولوشنی قادر به انتقال یادگیری به زمینه‌های مختلف یا داده‌های جدید نیستند و نیاز به آموزش مجدد برای هر مسئله ورودی دارند.**
8. **وابستگی شدید به موقعیت مکانی: شبکه‌های کانولوشنی برای داده‌هایی که رابطه مکانی بین آن‌ها وجود ندارد، به خوبی عمل نمی‌کنند.**
9. **زمان و هزینه محاسباتی: انجام کانولوشن در شبکه‌های کانولوشنی ممکن است زمان‌بر و هزینه‌بر باشد.**
10. **وابستگی شدید به حجم داده ورودی: شبکه‌های کانولوشنی به طور قابل توجهی به حجم داده ورودی برای آموزش و ترین نیاز دارند و برای بهبود کارکرد، تکنیک‌های مختلفی ابداع شده است.**
11. **کارآیی محدود به داده‌های تصویری: شبکه‌های کانولوشنی برای داده‌های غیر از تصویر قابلیت کارآیی محدودتری دارند و نمی‌توان آن‌ها را به‌طور کامل در کنار سایر معماری‌ها قرار داد و استفاده کرد.**